INL/EXT-05-00203

Programmers Manual for the PVM Coupling Interface in the RELAP5-3D© Code

Walter L. Weaver III

March 2005

INL/EXT-05-00203

Programmers Manual for the PVM Coupling Interface in the RELAP5-3D$^{©}$ Code

Walter L. Weaver III

March 2005

Idaho National Laboratory
Idaho Falls, Idaho 83415

Prepared Under DOE/NE Idaho Operations Office
Contract No. DE-AC07-05ID14517

ABSTRACT

This report describes the implementation of the PVM API in the RELAP5-3D© computer code. The information in the report is intended for programmers wanting to correct or extend RELAP5-3D©.

CONTENTS

1 Introduction

This document is the third in a series of reports that describe the Parallel Virtual Machine (PVM) coupling Application Programming Interface (API) and the code coupling system that was built using the PVM coupling API. The first report[1] describes the PVM coupling Application Programming Interface (API). The second report[2] describes the PVMEXEC program that controls and coordinates a coupled simulation. This report describes the implementation of the PVM coupling API in the RELAP5-3D© code.

1.1 Background

The PVM API and PVMEXEC program were developed to facilitate the simulation of a system (e.g., a nuclear power plant) using several different computer programs to describe the transient behavior of the system. Simulation codes are generally written to provide detailed models of some portion of a system, i.e., COBRA[3], RELAP5-3D[4], FLUENT[5], TRAC-PF1/MOD1[6], and TRACE[7] for the fluid systems, CONTAIN[8] and MELCOR[9] for the containment systems, NESTLE[10] and PARCS[11] for reactor power, etc. The PVM API, the PVMEXEC program, and the code coupling system that they implement enables the use of different codes for the simulation of different portions of the system in a unified analysis of the transient behavior of the system. Each code in the coupled simulation needs data from some of the others codes in the simulation. The data is passed between the several codes using the Parallel Virtual Machine[12] (PVM) message passing methodology. The PVMEXEC program was originally developed to couple the RELAP5-3D© code to other thermal-hydraulic codes, however any code that implements the PVMEXEC API may be used in a coupled simulation.

Several types of coupling have been implemented in the PVMEXEC program. These types of coupling can be categorized using three characteristics of the coupling, first by the computational models that are being coupled, second by the frequency of data exchanges between the computational tasks, and third by the type of solution algorithm being used by the coupling. The computational models that have been coupled are the thermal-hydraulics models, the kinetics models, and the control systems models. The second type of categorization yields synchronous coupling in which each task uses the same time step size and data is exchanged every time step and asynchronous coupling where data are exchanged at fixed intervals and the tasks are free to use their own time step sizes subject to the restriction of exchanging data at the correct time. Finally, the coupling may be categorized by the type of solution algorithm used by the coupling. In explicit coupling the data received from another task remains constant during the time step advancements. In semi-implicit coupling, some of the data received from the coupled task is advanced in time while some of the data is held constant during the time advancement. Not all types of coupling have been implemented in the PVMEXEC program. The types of coupling that have been implemented in the PVMEXEC program are synchronous and asynchronous explicit thermal-hydraulic coupling, semi-implicit thermal-hydraulic coupling (synchronous coupling by definition), synchronous kinetics coupling, and synchronous control systems coupling. The kinetics and control systems coupling are a form of explicit coupling because the data exchanged between the coupled codes remains constant while being used by the code that receives it. Explicit thermal-hydraulic coupling can be further subdivided into parallel explicit coupling and sequential explicit coupling. In parallel explicit coupling, the coupled tasks are peers and the order of the messages between them is arbitrary. In sequential explicit coupling, the leader task must be advanced first and then the follower task can be advanced using data received from the leader task. These roles define an order for the computations and data exchanges. There are roles and a defined order of computation and data exchanges for the other types of coupling. In semi-implicit coupling, the two roles are master and slave. The master task computes part of it's solution and sends the data from the partial advancement to the slave task that has been waiting for the data. The slave task uses the data received from the master task to complete it's advancement and sends it's final values back to the master task. The master task then uses the values received from the slave task to finish it's advancement. The advancement of the slave task is embedded in the middle of the master task.

The computational sequence for kinetics coupling is arbitrary where the thermal-hydraulic models can be advanced first using the power computed during the last coupled advancement and then the kinetics models can be advanced using the results of the thermal-hydraulic advancement or vise-versa, the kinetics solution may be advanced first, etc. In the kinetics coupling implemented in the PVMEXEC program, the thermal-hydraulic models are advanced first and then the kinetics models are advanced using the thermal-hydraulic conditions in the reactor core computed by the thermal-hydraulic advancement because this is the computational order for uncoupled computations in RELAP5-3D$^{©}$. The task performing the thermal-hydraulic advancement is designated as the client task and the task performing the kinetics solution is designated as the server task. The names client and server are used because the 'server' performs the work requested by the client.

Like kinetics coupling, the computational sequence for control systems coupling is arbitrary and the RELAP5-3D$^{©}$ computational order has been adopted by the PVMEXEC program, first the thermal-hydraulic models are advanced, then the kinetics models are advanced, and finally the control systems models are advanced. The order of the data exchanges for control systems coupling are determined by the user in the numbering of the control components. The control components are advanced in numerical order as specified by the user. Care must be taken in numbering the control components so that the computational order of control components in one task that are to send data is the same as the computational order of the control components in the other tasks that are to receive data and vise-versa.

RELAP5-3D$^{©}$ has been modified so that it can function in any role in all types of coupling, either as the master task or as the slave task in semi-implicit thermal-hydraulic coupling, as either the' serve'r task or 'client' task in kinetics coupling, and as the leader task or the follower task in explicit sequential thermal-hydraulic coupling. The tasks in the other types of coupling are peers and there is no defined role for these types of coupling.

This report is intended as a programmers manual and describes the changes that have been made to the RELAP5-3D$^{©}$ code to allow it to be part of a coupled simulation. The original implementation of a PVM coupling interface[13] was removed from the code and its control functionality was reproduced in the PVMEXEC code. This made the PVM coupling methodology independent of RELAP5-3D$^{©}$. The original PVM coupling interface only implemented explicit parallel thermal-hydraulic coupling and new types of coupling were developed and implemented in the PVMEXEC code. The reference implementation of these new types of coupling, i.e., semi-implicit thermal hydraulic coupling, kinetics coupling, and control systems coupling, as well as enhanced explicit thermal-hydraulic coupling, in the new PVM coupling system was in the RELAP5-3D© code.

2 Code Modifications

The implementation of the PVM coupling methodology in RELAP5-3D© required the modification of 87 subroutines and comdecks. The modifications to the subroutines and comdecks are delimited by the preprocessor symbol ‘pvmcoupl’. The modifications to subroutines are described first (in alphabetical order), and then the modifications to comdecks. The intent of the description of the modifications is to describe WHAT the modifications are intended to accomplish rather than to describe HOW the modifications perform their functions. It is also assumed that the reader of this report has access to the source code for the RELAP5-3D© program so that the details of HOW the modifications perform their function can be determined by reading the coding.

It is difficult to describe the changes that have been made to the code to implement the PVM coupling methodology. One of the problems is that the changes for the different types of coupling are intermixed within the subroutines. Another problem is that the changes for a successful time advancement are intermixed with the changes for an unsuccessful time step. The changes for an unsuccessful advancement are different depending upon what type of failure caused the failure of the advancement. A advancement may fail for several reasons like excessive mass error, a fluid property error, velocity flip-flop, water packing, appearance of non condensable gas, etc. The recovery strategy is different for each of these types of failure and the modifications for recovery for PVM coupled problems are different for each type of failure. Finally, the changes are hard to describe because they modify the solution algorithm for uncoupled simulations and without an understanding of how the uncoupled algorithm works it is hard to describe how it has been modified.

2.1 Subroutine ADECHK

Subroutine ADECHK was modified to add a printout of the status of the new preprocessor symbol ‘pvmcoupl’ that controls the addition of the PVM coupling modifications to the executable file for RELAP5-3D©.

2.2 Subroutine BLKDTA

Subroutine BLKDTA was modified to add default values for several PVM coupling variables. The variables added to RELAP5-3D© for the implementation of the PVM coupling methodology are listed in Appendix A along with a short description of each variable.

2.3 Subroutine CONVAR

Subroutine CONVAR was modified to define a new type of control component, the coupling component. The coupling component can send its input to another coupled task, can define its output from a value received from a coupled task, or send its input value and then receive its output value from another task participating in control system coupling.

2.4 Subroutine CPLFNCTN

Subroutine CPLFNCTN implements the coupling control component. It sends its input value by calling subroutine PVMSND with a value of five for the input parameter, then receives its output value by a call to subroutine PVMRCV with a input parameter value of five, followed by a call to subroutine PVMPUT with a input parameter value of five.

2.5 Subroutine DTSTEP

Subroutine DTSTEP was modified extensively for the implementation of PVM coupling. The modifications are for the control of the production of printed output, plottable output, and restart information, for the control of the data exchanges for explicit coupling, and for the determination of the time step size when RELAP5-3D© is participating in synchronous coupling.

The modifications for the control of output include both the determination that output should be produced this time step and the determination of what output should be produced. Output is produced whenever the current time is equal to one of a set of times, one time for the production of minor edits, one for writing information for the plot file, one time for the production of major edit, and one time for writing restart information to the restart file. These times are determined from input data on the time step cards for uncoupled runs. For coupled simulations, these times are either received from the PVMEXEC program or are determined from the time step data in the RELAP5-3D© input file. The minor edit times, the plot times, the major edit times, and the restart times are determined by the PVMEXEC program for synchronous coupling while only the restart times are determined by the PVMEXEC program for asynchronous coupling. The other times for asynchronous coupling are determined as for uncoupled simulations. Subroutine DTSTEP determines if output should be produced at the current simulation time and produces output as appropriate.

The next set of modifications was for the exchange of data for explicit coupling. These data exchanges are similar to the data exchanges performed during the initialization phase of a coupled computation and are under the control of the PVMEXEC program. If RELAP5-3D© is participating in synchronous, parallel explicit coupling, subroutine DTSTEP listens to receive messages with message tag 8002 from the PVMEXEC program. These messages contain the task identifier of the task that is to send its data to the other tasks participating in synchronous parallel explicit coupling. If the task identifier contained in the message is the same as the task identifier of this instance of RELAP5-3D©, subroutine DTSTEP calls subroutine PVMSND, otherwise it calls subroutine PVMRCV and PVMPUT. Once the called subroutines return, it sends a message with message tag 8002 containing its task identifier to the PVMEXEC program to signify that it has finished exchanging data with the task specified in the original 8002 message. It continues to listen for and process messages with message tag 8002 until it receives a message containing a value of zero. This designates that all of the data exchanges for synchronous parallel explicit coupling have been performed.

Subroutine DTSTEP next listens to receive messages with message tag 8003 from the PVMEXEC program if this instance of RELAP5-3D© is participating in asynchronous parallel explicit coupling and the current time is the end time of the explicit coupling interval (the end time of the explicit coupling interval is received from the PVMEXEC program as will be explained later). Under these conditions, subroutine DTSTEP repeats the same procedure as explained in the previous paragraph except the control messages from the PVMEXEC program are messages with message tag 8003.

Next, subroutine DTSTEP listens to receive message with message tag 8004 if this instance of RELAP5-3D© is participating in synchronous sequential explicit coupling. The messages with message tag 8004 are processed in the same way as the messages with message tag 8002 and 8003.

Finally, subroutine DTSTEP listens to receive messages with message tag 8005 from the PVMEXEC program if this instance of RELAP5-3D© is participating in asynchronous sequential explicit coupling and the current time is at the end of the explicit coupling interval. The processing of messages with message tag 8005 is the same as for messages with message tag 8002, 8003, and 8004.

After the production of output and the exchange of explicit coupling data, subroutine DTSTEP determines if any of these times need to be updated. If the current simulation is equal to any of the output times determined by the PVMEXEC program or is equal to the end of the explicit coupling interval, subroutine DTSTEP listens to receive a message with message tag 8000. This message contains a new set of output times and a new explicit coupling interval time. Also included in this message is a new value of the end time of the coupled simulation and the maximum and minimum time step sizes for the coupled simulation. The maximum and minimum time step sizes are only used if this instance of RELAP5-3D© is participating in synchronous coupling. Some to these values may be the same as the previous value for an output time because that time for the production of that type of output has not yet been reached. In any case, one of more of these times will be different.

Subroutine DTSTEP then determines the time step size that it wishes to use for the next advancement. This time step might be a reduced time step because of a time step advancement failure, might be a reduced time step because of Courant limit violations, might be a larger time step size because of low mass error, etc. After the new time step size is determine, it is sent to the PVMEXEC program if this instance of RELAP5-3D© is participating in synchronous coupling. It sends its desired time step size in a message with message tag 8001 and listens to receive a message back from the PVMEXEC program with the same message tag. This message will contain the global time step size for synchronous coupling but also a new set of output times for minor edit, plots and major edits. These times are included with the global time step size so that output might be produced for every successful time advancement.

2.6 Subroutine EQFINL

Subroutine EQFINL was modified to use the flow rates in the coupling junctions received from the slave task in the conserving step for the mass and energy equations for PVM coupling volumes in the master task for semi-implicit thermal-hydraulic coupling. The control logic for the appearance of noncondensable gas was modified to use the variables that control repeated time steps for coupled simulations. The default values of the PVM variables ensure that the logic is correct if RELAP5-3D© is executed in uncoupled mode as a stand-alone program.

2.7 Subroutine GETSEC

Subroutine GETSEC computes the time in seconds from the values returned by the system timing routine. This routine is not used in RELAP5-3D©.

2.8 Subroutine GNINIT1

Subroutine GNINIT1 was modified to determine if the parameter '-PVM' occurs on the command line. If this parameter occurs on the command line, it indicates that RELAP5-3D© has been executed under the control of the PVMEXEC program. It then determines the task identifier of its parent task using a call to the PVM library routine 'pvmfparent'. Once it knows the task identifier of its parent, it listens to receive a message with message tag 7001 from its parent. After this message has been received and acknowledged, it sets the task identifier of the PVMEXEC task from the contents of message 7001. It asks the PVM virtual machine to send it a message with message tag 10001 if the PVMEXEC program fails by calling the PVM library routine 'pvmfnotify'.

2.9 Subroutine HT1SST

Subroutine HT1SST has been modified to read the power in a heat structure from the data received from a coupled task if RELAP5-3D© is the client task in kinetics coupling (RELAP5-3D© can be either the client or the server task in kinetics coupling).

2.10 Subroutine HT1TDP

Subroutine HT1TDP has been modified to read the power in a heat structure from the data received from a coupled task if RELAP5-3D© is the client task in kinetics coupling.

2.11 Subroutine HT2TDP

Subroutine HT2TDP has been modified to read the power in a heat structure from the data received from a coupled task if RELAP5-3D© is the client task in kinetics coupling.

2.12 Subroutine HTADV

Subroutine HTADV has been modified to bypass the computation of the temperatures in a heat structure if the heat structure is connected to a PVM coupling volume in the slave task for semi-implicit thermal-hydraulic coupling. The surface temperature of a heat structure connected to a PVM coupling volume is set using data received from the corresponding master task if the master task is another instance of RELAP5-3D©. This data exchanges was added to the semi-implicit coupling so that the results of a coupled simulation using RELAP5-3D© for both the master and the slave tasks would produce the same results as an uncoupled simulation of the same problem. This was done for verification of the coding for semi-implicit thermal-hydraulic coupling. The surface temperature is used in the computation of the heated wall effect on the friction factor in the PVM coupling volume in the slave task.

2.13 Subroutine HTCOND

Subroutine HTCOND has been modified to used the value from a general table as the boundary condition for the heat structure where the value from the table is set from data received from the explicitly coupled task.

2.14 Subroutine HTFINL

Subroutine HTFINL has been modified to bypass the back substitution step for heat structures that are coupled semi-implicitly to their sink volume if the sink volume is PVM coupling volume in the slave process. The forward elimination step is performed in subroutine HTADV and is also bypassed if the sink volume for the heat structure is a PVM coupling volume in the slave process.

2.15 Subroutine HYDRO

Subroutine HYDRO has been modified to implement several phases of semi-implicit thermal-hydraulic coupling. First, subroutine HYDRO listens at the beginning of the computation to receive the volume average fluid velocities and heat structure surface temperatures for PVM coupling volumes if this instance of RELAP5-3D© is serving as the slave task for semi-implicit thermal-hydraulic coupling. At the end of a successful advancement, subroutine HYDRO listens to receive the fluid properties in PVM coupling volumes if this instance of RELAP5-3D© is serving as the slave process in semi-implicit thermal-hydraulic coupling before computing the fluid properties in the volumes. After the computation of the fluid properties, subroutine HYDRO will send the properties in PVM boundary volumes to the appropriate slave task in this instance of RELAP5-3D© is serving as the master task in semi-implicit thermal-hydraulic coupling. after the determination of the junction properties from the volume properties, the junction properties are send from the slave process to the master process for PVM coupling junctions and are received by the master process from the slave process (the computation of PVM coupling junction properties are bypassed in the PVM master process). Finally, instead of the data exchanges that are performed at the end of a successful advancement, the exchange of fluid properties in the PVM coupling volumes and junctions is performed if the time step is to be repeated with the same time step size (i.e., for velocity flip-flop, water packing, or the appearance of non-condensable gases).

2.16 Subroutine ICMPN1

Subroutine ICMPN1 has been modified initialize the several PVM coupling flow rate variables and the single PVM coupling volume variable. Semi-implicit thermal-hydraulic coupling defines several new flow rate variables for junctions needed by the semi-implicit coupling solution algorithm. The flow rates are the mass flow rates of liquid and vapor, the flow rate of liquid internal energy, the flow rate of vapor internal energy, the volumetric flow rates of liquid and vapor, and the total mass flow rate of non condensable gas.

2.17 Subroutine ICOMPN

Subroutine ICOMPN has been modified to perform data exchanges if this instance of RELAP5-3D© is participating in explicit thermal-hydraulic coupling. Subroutine ICOMPN first marks the PVM coupling volumes and junctions depending upon whether a volume or junction is specified in an explicit receive message or not. Then subroutine ICOMPN computes the initial values of the fluid properties in the volumes and the initial velocities in the junctions that are not PVM coupling components by calls to subroutine ISTATE and subroutine IVELST respectively with an input parameter of zero. The input parameter of zero indicates that the computations for the PVM coupling components are to be bypassed. Subroutine ICOMPN then listens to receive a series of data exchange control messages from the PVMEXEC program in messages with message tag 8002. Each 8002 message contains the task identifier of one of the tasks participating in explicit parallel thermal-hydraulic coupling. If the task mentioned in the 8002 is the task identifier of this instance of RELAP5-3D©, subroutine ICOMPN sends its data to the other tasks that are participating in explicit synchronous parallel thermal-hydraulic coupling. After sending all of its messages and receiving an acknowledgement for each message, it sends a message to the PVMEXEC program in a message with message tag 8002 containing its task identifier. If the task mentioned in the 8002 message is not its task identifier, it listens to receive data from the task specified in the 8002 message if it is to receive data from the specified task. It listens, receives data and sends acknowledgements to the sending task until all the data has been received from the specified sending task (or no data has been received from the specified task because no data exchanges between these two tasks has been specified) and then sends a message to the PVMEXEC program in a message with message tag 8002 containing its task identifier. This acknowledgement to the PVMEXEC program signifies that it is finished exchanging data with the specified task. It then listens to receive the next message with message tag 8002 from the PVMEXEC program. If the message from the PVMEXEC program contains a task identifier of zero, it acknowledges the message and listens for a series of messages with message tag 8003. A message containing a task identifier of zero denotes that the data exchanges for explicit synchronous parallel thermal-hydraulic coupling are finished. The control messages in messages with message tag 8003 are for tasks participating in explicit asynchronous parallel thermal-hydraulic coupling. The procedure for sending and receiving data for explicit asynchronous parallel thermal-hydraulic coupling coordinated by the 8003 messages is the same as the procedure for data exchanges coordinated by messages with message tag 8002.

After the sequence of 8003 messages from the PVMEXEC program, subroutine ICOMPN listens to receive a series of messages with message tag 8004 from the PVMEXEC program. These messages coordinate the sending and receiving of data for explicit synchronous sequential thermal-hydraulic coupling. The series of 8004 messages are received from the PVMEXEC program twice, the first sequence of messages coordinates the exchange of volume data and the second series of messages with message tag 8004 from the PVMEXEC program coordinates the exchange of junction data.

After the two series of 8004 messages, subroutine ICOMPN listens to receive a series of messages with message tag 8005 from the PVMEXEC program. These messages coordinate the exchange of data for explicit asynchronous sequential thermal-hydraulic coupling. As with the 8004 messages, the 8005 messages are received twice, the first time for the exchange of volume data and the second for the exchange of junction data.

After all of the data exchanges of explicit thermal-hydraulic coupling have been performed, subroutines IJPROP and IVELST are called with an input parameter of one signifying that the PVM coupling volumes and junctions are to be initialized using the data received during the initial data exchanges. This completes the data exchanges for explicit thermal-hydraulic coupling.

Further down in subroutine ICOMPN, data exchanges are performed for semi-implicit thermal-hydraulic coupling. These data exchanges are nested around the call to subroutine IJPROP, the subroutine that initializes junction properties from volume properties. If this instance of RELAP5-3D© is functioning as the master process in semi-implicit thermal-hydraulic coupling, it sends the PVM coupling volume properties to it’s slave process before calling subroutine IJPROP to initialize the properties in it’s junctions that are not PVM coupling junctions. After computing the properties in junctions that are not PVM coupling junctions, it listens to receive the junction properties in the junctions that are PVM coupling junction from it’s slave process. The backup flag is set to -1 before the PVM receive routine is called so that once it stores the received values in the ‘new’ storage locations in the PVM internal file it copies them to the ‘old’ storage locations. The backup flag is then set to zero (this indicates that the previous advancement, i.e., the nonexistent advancement, was successful). This completes the initialization of junction properties for the master process in semi-implicit thermal-hydraulic coupling. If the process is the slave process in semi-implicit thermal-hydraulic coupling, it listens to receive the properties in its PVM coupling volumes from its master process after setting the backup flag as described above. After receiving the properties in the coupling volumes and resetting the backup flag, it computes the properties in its junctions by calling subroutine IJPROP. Subroutine IJPROP initializes the properties in all of its junctions. Once subroutine IJPROP has computed the properties in all of the junctions in the slave process, it sends the properties in the PVM coupling junctions to its master process. This completes the data exchanges for semi-implicit thermal-hydraulic coupling and the modifications to subroutine ICOMPN for PVM coupling.

2.18 Subroutine ICONVR

Subroutine ICONVR has been modified to call subroutine CPLFNCTN if the current control component is a PVM coupling control component. Subroutine CPLFNCTL performs data exchanges to initialize the control system database for PVM coupling components.

2.19 Subroutine IHSEFL

Subroutine IHSEFL initializes the PVM coupling flow rate variables for junctions using the pull-thou model.

2.20 Subroutine IHTCMP

Subroutine IHTCMP initializes the pointers to the power data for the heat structures in the instance of RELAP5-3D© serving as the client task in kinetics coupling. The power in the heat structures are computed in the server task and are sent to the client task. The power data is stored in the coupling database of the client task because there is no power database (i.e., internal file) in the client task.

2.21 Subroutine IHTWRD

Subroutine IHTWRD was modified to bypass the initialization of heat structures connected to PVM coupling volumes in the slave task for semi-implicit coupling.

2.22 Subroutine IHZFLW

Subroutine IHZFLW was modified to compute the several PVM flow rate variables for junctions using the pull-three model.

2.23 Subroutine IJPROP

Subroutine IJPROP was modified so that the initialization of junction properties is bypassed if a junction is a PVM coupling junction in the follower process for explicit sequential thermal-hydraulic coupling.

2.24 Subroutine IMLP

Subroutine IMLP was modified to combine separate computational systems into a single computational system if any of the PVM coupling components in any of the computational systems are coupled to the same task. This combining of computational systems is only performed for semi-implicit thermal-hydraulic coupling.

2.25 Subroutine INPUTD

Subroutine INPUTD was modified to send the initialization and run status of RELAP5-3D$^{©}$ to the PVMEXEC program in a message with message tag 9000. Subroutine INPUTD then to listens to receive the global run status from the PVMEXEC program in a message with message tag 9000. If the global run status is stop, the failure flag is set to true so that RELAP5-3D$^{©}$ will terminate gracefully.

2.26 Subroutine IR5PVMC

Subroutine IR5PVMC was added to RELAP5-3D$^{©}$ as part of the original implementation of explicit parallel thermal-hydraulic coupling. It has been modified extensively for the new types of coupling and for use with the PVMEXEC program. Subroutine IR5PVMC processes the metadata for each type of message and sets the pointers to the data items in the RELAP5-3D$^{©}$ internal files by calls to subroutine SCNREQ. Subroutine SCNREQ returns two data items, the internal file number, and the offset in the internal file to the data item. These two values are stored in the metadata for each data item that is to be sent or received. The pointers in the metadata for send data items are used to load the value of the data item from the RELAP5-3D$^{©}$ internal files into the send buffer. The values of the pointers in the metadata for the receive messages are used by subroutine PVMPUT to load the values from the receive buffer into the RELAP5-3D$^{©}$ internal files so that they can be used in the numerical solution algorithm. The flow rates of non condensable gases in coupling junctions are converted to the mass qualities in the junction (because the RELAP5-3D$^{©}$ solution algorithm uses mass quality instead of mass flow rate in the solution for the continuity equations for non condensable gases. This means that the variable name in the metadata for the flow rate of non condensable gas is converted into the appropriate name for mass quality before subroutine SCNREQ is called for these data items. After processing the direct send and receive messages for semi-implicit coupling, the indirect send messages for the slave task, and the indirect receive messages for the master task, the system index is set for every message, and the master/slave flag, and the task identifier of the coupled task are set for every computational system. The coupling junction number for each of the coupling volumes in the master task for semi-implicit thermal-hydraulic coupling are set and the number of coupling volumes is set for each computational system. Next the connection codes for the connection of coupling junctions to coupling volumes is checked to ensure that the coupling volume is the ‘to’ volume for the coupling junction. This check is made because a positive value of the coupling junction flow rate is defined as into the coupling volume in the master task in semi-implicit coupling. The coupling volumes are also checked to ensure that each coupling volume has a coupling junction connected to it.

After the semi-implicit messages have been processed, the kinetics messages are processed. The pointers are set for the different types of data items that can be sent in kinetics send messages. This process is the same as for the variables in explicit messages and direct messages for semi-implicit coupling. The pointers for receive messages are different because the data in kinetics receive messages are stored in internal file 31 instead of in the regular internal files. This is needed because the data are for components that do not exist in the system model for the task. In the client task, the power data in stored in file 31 because the power model cannot be used in the client task (i.e., the power is to be computed by the server task). Conversely, the volume and heat structure data needed by the server task for the computation of the reactor power are computed in the client task and these volumes and heat structures are not part of the system model in the server task. The data items are stored in file31. The power data items are stored in a block of five words at the end of the metadata that define the power data. The heat structure data are stored as a single word at the end of the metadata that defines heat structure data terms. The volume data are stored in a block of 17 words at the end of the metadata for a volume. The pointers are set so that the locations of the data items in the data block have the same relative location to each other that they have in the VOLDAT common block. This is done so that the references to these data items in the kinetics routine can use the same equivalence methodology as for volume variables in the VOLDAT common block.

The pointers for control system data items are set in the same manner as for data items in explicit messages. This finishes the processing in subroutine IR5PVMC.

2.27 Subroutine IRKIN

Subroutine IRKIN was modified to set the pointers used in the computation of the fluid properties in the reactor core to point to the PVM internal file (file 31) instead of the volume database in the server task for kinetics coupling. The data that is received from the client task in kinetics coupling cannot be stored in the volume database because the properties are for volumes that do not exist in the server task (the fluid properties in some [or all] of the volumes in the reactor core are computed by the client task). These properties are stored in the coupling database (file 31) and the pointers are set accordingly so that the kinetics routines that compute the fluid properties in the reactor core do not need to be modified. The pointers to heat structure data are also set appropriately. The pointer for volume data is decremented by 30 (the relative location of the first word in the PVM internal data file [variable voidf] relative to the start of volume data in the voldat file. This is done so that the equivalence methodology can be used to reference the data without needing to modify the kinetics routines that use the pointers.

2.28 Subroutine ISTATE

Subroutine ISTATE has been modified to bypass the initialization of PVM coupling volumes if the input control flag is zero (i.e. the first pass) and to bypass the initialization of volumes that are not PVM coupling volumes if the input control flag is one (i.e., second pass). This logic is used to control the initialization of PVM coupling volumes for explicit thermal-hydraulic coupling. The independent variables for PVM coupling volumes are set by a call to subroutine PVMSET. This routine loads the ‘prop’ array from the data received from the coupled task. The ‘prop’ array is used to transfer the independent variables that are to be used in the computation of the fluid properties to the appropriate steam table routines based on the fluid type. The computed properties are returned to subroutine ISTATE in the ‘prop’ array and are loaded into the volume database for the particular volume by subroutine ISTATE.

2.29 Subroutine IVELST

Subroutine IVELST has been modified to bypass the initialization of the fluid velocities in PVM coupling junctions that receive their data from a coupled task if the value of the input control flag is zero (i.e., the first pass) and to bypass the initialization of the fluid velocities in junctions that are not PVM coupling junctions if the input control flag in one (i.e., the second pass). The ‘prop’ array used in the initialization of the junction velocities for PVM coupling junctions is set by a call to subroutine PVMSETJ that loads the ‘prop’ array from the data received from the coupled task. Once the ‘prop’ array is loaded correctly for PVM coupling junctions, the computation of the fluid velocities in the junctions proceeds as for junctions that are not PVM coupling junctions.

2.30 Subroutine JPROP

Subroutine JPROP has been modified to bypass the velocity flip-flop check for junctions that are PVM coupling junctions in the master process for semi-implicit thermal-hydraulic coupling or are PVM coupling junctions in the follower task for explicit sequential thermal-hydraulic coupling. The velocity flip-flop check for these junction must be done in the other task since the velocity in these junctions is being computes by the other task.

2.31 Subroutine MOVER

Subroutine MOVER has been modified to execute the backup logic for junctions if: a) the task is not synchronously coupled to another task, and the junction is not a time dependent junction or the time step is to be repeated with a smaller time step, or b) if the task is synchronously coupled to another task. The backup logic for volumes has been modified so that the volume variables for noncondensable gas are returned to their initial values if the time step is to be repeated with the same time step size, and a) the task is not synchronously coupled to another task, or b) the task is synchronously coupled to another task and the repeated time step was caused by the appearance of noncondensable gas. If the task is participating in semi-implicit coupling, the PVM backup flag is set to one and subroutines PVMRCV and PVMPUT are called to restore the values in the coupling components to their beginning of advancement values. The PVM backup flag is reset to a value of zero after subroutine PVMPUT finishes.

2.32 Subroutine PLTREAD

Subroutine PLTREAD was modified to read the several junction flow rate variables used by semi-implicit thermal-hydraulic coupling from the plot records. Although the variables are only used by PVM coupling junctions in semi-implicit coupling, their values are computed by the code for all junctions and for all types of coupled and uncoupled simulations.

2.33 Subroutine PLTWRT

Subroutine PLTWRD was modified at write the values of the several junction flow rate variables used by semi-implicit thermal-hydraulic coupling to the default section of the plot file. Although the variables are only used by PVM coupling junctions in semi-implicit coupling, their values are computed by the code for all junctions and for all types of coupled and uncoupled simulations.

2.34 Subroutine PRESEJ

Subroutine PRESEJ was modified to bypass the addition of the contribution of the flux terms to the solution matrix for the mass and energy equations for PVM coupling junctions in the master task for semi-implicit thermal-hydraulic coupling. The subroutine was modified to add the contribution of the flux terms to the right hand side of the solution system for junctions that are PVM coupling junctions in the follower task for explicit sequential thermal-hydraulic coupling. The values of the several flow rate used in these terms are computed using data received from the corresponding leader task. Finally, the subroutine was modified to add the implicit contributions to the solution matrix and the constant terms to the right hand side of the flux terms in the mass and energy equations for PVM coupling junctions and PVM coupling volumes in the slave process for semi-implicit thermal-hydraulic coupling.

2.35 Subroutine PRESEQ

Subroutine PRESEQ was modified to add the constant term to the pressure equation for PVM coupling volumes in the slave task in semi-implicit thermal-hydraulic coupling. The constant term is computed by the corresponding master task and is received from that task in a PVM message. It then zeros out the extra right hand side storage locations for the implicit part of the pressure equations for PVM coupling volumes in the slave task. It then adds the contribution of the implicit part of the pressure equations for PVM coupling volumes in the slave task to the appropriate right hand side location using the data received from its master task.

2.36 Subroutine PVMFXREC

Subroutine PVMFXREC was added to RELAP5-3D$^©$ for the original implementation of explicit thermal-hydraulic coupling. It has been modified extensively for the new types of PVM coupling. Subroutine PVMFXREC is the routine called to received messages from the other simulation tasks that are executing on the virtual machine. Each message has a wait time. It divides the wait time into a number of wait intervals and calls the PVM library blocking routine ‘pvmftrecv’ to receive a message with the specified message tag from the specified task. If the blocking routine receives the specified message at any time during the wait interval, it returns with a error status of zero. PVMFXREC checks the errors status and returns if the error is zero. If the message has not been received during the wait interval, subroutine PVMFXREC checks to determine if a message with message tag 10003 has been received from the PVMEXEC program by a call to PVM library non blocking routine ‘pvmfnrecv’. This routine returns immediately regardless of whether the specified message has been received or not received. If such a message has been received, the error flag is set appropriately and the routine PVMFXREC returns to the calling routine. A message with message tag 10003 indicates that one of the simulation tasks has either exceeded its wait time while waiting to receive a message from another simulation task or has failed. In any case, a message with message tag 10003 denotes an error condition and indicates that the coupled simulation is to be terminated. A task receiving a message with message tag 10003 is expected to terminate itself gracefully, i.e., write final output, etc., and then stop. If a message with message tag 10003 has not been received, it checks to see if a message with message tag 10001 has been received from the virtual machine. If such a message has been received (denoting that one of the PVMEXEC program has terminated unexpectedly), it sets the error flag appropriately and returns to the calling routine. Finally, it checks to see if the number of wait intervals has been exceeded. If the number of wait intervals has been exceeded, the error flag is set to the negative of the task identifier of this instance of RELAP5-3D$^{©}$, a message with message tag 10002 containing the error flag, the message tag and the task identifier from which the message was to be received and the error flag 10002 is sent to the PVMEXEC program, and routine PVMFXREC returns to the calling routine. If the number of wait intervals has not been exceeded, it returns to the top on the routine and begins the next wait cycle by calling the PVM library blocking routine ‘pvmftrecv’. The routine continues to cycle until either a message has been received or the number of wait cycles has been exhausted. The error flag values returned by subroutine PVMFXREC are listed in Appendix B.

2.37 Subroutine PVMPUT

Subroutine PVMPUT is called immediately after subroutine PVMRCV to transfer the data in the receive buffer or PVM internal file into the appropriate storage locations in the RELAP5-3D$^{©}$ database (i.e., the other internal files). The subroutine is divided into seven sections. The particular section to be executed is specified in the subroutine input parameter.

If the input parameter is zero, data for explicit parallel thermal-hydraulic coupling has been received. The data must be loaded into the RELAP5-3D$^{©}$ database using the pointers stored in the metadata for the message. The subroutine first loops over all explicit messages looking for explicit parallel coupling messages that are to be received by this task. Having found an appropriate explicit parallel message, it writes the values in the receive buffer into the appropriate storage locations in the RELAP5-3D$^{©}$ database (i.e., the internal files). If the variable is a flow rate of a noncondensable gas, the flow rate is converted into a mass quality by dividing the mass flow rate of the particular noncondensable gas by the total flow rate of noncondensable gases before storing the value in the RELAP-5-3D$^{©}$ internal file.

If the value of the input parameter is one, the message contains data for the direct semi-implicit thermal-hydraulic coupling messages. The subroutine finds the semi-implicit message for the current computational system and loads the data from the ‘new’ storage locations in the PVM internal file into the appropriate storage locations in the RELAP-53D$^{©}$ database.

If the input parameter is two, the message contains data for the indirect semi-implicit coupling messages (there can be up to three different types of messages for semi-implicit coupling). The message database is searched for the appropriate message and the data loaded from the receive buffer into the RELAP5-3D$^{©}$ internal files using the same procedure used for messages with control flags of zero. Just as in the processing for messages with control flag of zero, the mass flow rate of a noncondensable gas is converted into the mass quality before being stored.

If the input parameter is three, the message is the third type of message for semi-implicit coupling. The subroutine loops over the explicit message metadata to determine if a message is to be received from the sending task designated by the PVMEXEC program. If a message is to be received from the currently designated sending task, the name of the sending task is compared to the string ‘relap5.x’. If the name of the sending task matches the string, the data is stored as detailed previously. If the name does not match the string, the next message is processed until all explicit receive messages have been processed.

If the input parameter is four, kinetics data have been received. The message database is searched for the appropriate message. The type of data that has been received in the message is determined for the metadata for the messages and the appropriate number of data items from the receive buffer are stored in the RELAP5-3D© database.

If the input parameter is five, control system coupling data have been received. The data is stored into the appropriate storage locations in the RELAP5-3D$^{©}$ database.

If the input parameter is six, volume data for explicit sequential thermal-hydraulic coupling have been received. It is stored into the RELAP5-3D$^{©}$ database using the procedure outlined in the previous paragraphs.

Finally, if the input parameter is seven, junction data for explicit sequential thermal-hydraulic coupling have been received. It is stored into the RELAP5-3D$^{©}$ database using the procedure outlined in the previous paragraphs

2.38 Subroutine PVMRCV

Subroutine PVMRCV is the subroutine that calls PVMFXREC to receive messages. It is divided into seven sections. The particular section to be executed is specified in the subroutine input parameter.

If the input parameter is zero, explicit parallel thermal-hydraulic messages are to be received. The number of explicit messages to be received is determined from the PVM internal file and a loop over explicit receive messages is executed. The loop first determines the whether the particular message is for parallel or sequential explicit thermal-hydraulic coupling by examining the explicit type flag. If the current message is for parallel explicit coupling (i.e., the explicit type flag for the message is negative one), the message tag and synchronous type for the message are determined from the PVM internal file. Subroutine PVMFXREC is called if the current message is to be received from the task designated as the sending task by the PVMEXEC program and the synchronous type of the message is the same as the synchronous type designated by the PVMEXEC program. Synchronous and asynchronous messages are received by separate calls to subroutine PVMRCV. After PVMFXREC returns, subroutine PVMRCV returns if the error flag is not zero or unpacks the data items from the PVM receive buffer into the RELAP5-3D$^{©}$ receive buffer. After unpacking the data items into the local receive buffer, it sends the message acknowledgement to the sending task. The processing of explicit messages continues until all explicit messages have been processed.

If the input parameter is one, a single direct semi-implicit thermal-hydraulic coupling message is to be received. The PVM internal file is examined to find the appropriate receive message for this computational system and the message tag and task identifier of the sending task are determined. Then subroutine PVMFXREC is called to receive the message. After subroutine PVMFXREC has returned, the error flag is checked for an error condition. If an error has occurred, subroutine PVMRCV returns. If the error flag is zero, the data is unpacked into the 'new' storage locations in the PVM internal and an acknowledgement is send to the sending task. There is no looping over messages because there can be only one receive message per computational system for semi-implicit thermal-hydraulic coupling.

If the input parameter is two, a second type of semi-implicit thermal-hydraulic coupling message is to be received. The procedure for receiving the message is the same as the procedure for receiving semi-implicit messages for an input parameter of one except that a different section of the PVM internal file is used to determine the message tags and sending task identifiers. The data is unpacked into the receive buffer instead of the PVM internal file.

If the input parameter is three, a third type of message for semi-implicit thermal-hydraulic coupling message is indicated. The PVM internal file is examined to determine if the current computational system is to receive semi-implicit coupling data. If the computational system is to receive data, the name of the sending task is determined from the task identifier of the sending task by a call to PVM library routine 'pvmftasks'. If the name of the sending task in not 'relap5.x', subroutine PVMRCV returns. If the name of the sending task is 'relap5.x', the data is received using the same procedure as for receiving data when the input flag has a value of one or two. These messages are only received by RELAP5-3D$^{©}$ if two instances of RELAP5-3D$^{©}$ are coupled to each other using semi-implicit coupling. These messages were included so that the results of coupling two instances of RELAP5-3D$^{©}$ to each other could be compared to the results of the simulation of the same problem using a single instance of RELAP5-3D$^{©}$ executed in uncoupled mode. The results of these two simulations were used to verify that the implementation of semi-implicit thermal-hydraulic coupling was working correctly.

If the input parameter is four, kinetics messages are to be received. The PVM internal file is examined to determine the number of messages that are to be received for kinetics coupling and a loop over messages is executed. Each message is received by a call to subroutine PVMFXREC and the data received is unpacked into the receive buffer. An acknowledgement is sent to the sending task before proceeding to the next message.

If the input parameter in five, control system coupling messages are to be received. The procedure for control system messages is the same as for kinetics messages except that a different portion of the PVM internal file is used to determine the message tag and the task identifier of the sending task. The numerical identifier of the control component calling subroutine PVMRCV is compared to the numerical identifier of the control component in the PVM internal file for the current message before calling subroutine PVMFXREC to actually receive the message. The data received is unpacked into the receive buffer and the acknowledgement is sent before proceeding to the next message in the section of the PVM internal file for control system messages.

If the input parameter is six, messages containing volume data for the leader task of explicit sequential thermal-hydraulic coupling are to be received. The message specification data for sequential explicit thermal-hydraulic messages are stored in the same portion of the PVM internal file as the specifications for messages for explicit parallel thermal-hydraulic coupling. The number of explicit messages is determined from the section of the PVM internal file for explicit messages and a loop over all explicit messages is executed. The explicit type for each message is examined and the message is received if the explicit message type flag is zero indicating that the message is for the leader task in explicit sequential thermal-hydraulic coupling. The message is received by a call to subroutine PVMFXREC if the task identifier of the task sending the message is the same as the task identifier of the task specified as the current sending task by the PVMEXEC program. The sending and receiving of all explicit message of whatever type is coordinated by the PVMEXEC program. Once received, the data is unpacked into the local receive buffer and the acknowledgment is sent to the sending task. The loop over explicit message continues until all explicit messages have been processed.

Finally, if the input parameter is seven, messages containing junction data for the follower task in explicit sequential thermal-hydraulic coupling are to be received. The procedure is the same as for receiving data for the leader task except that the explicit type flag must be one instead of zero.

2.39 Subroutine PVMSET

Subroutine PVMSET is the routine that loads the ‘prop’ array for the computation of fluid properties in time dependent volumes that are PVM coupling volumes. The values loaded into the ‘prop’ array are obtained from the data received in PVM messages and stored in the RELAP5-3D$^{©}$ or PVM databases. The ‘prop’ array is loaded for regular time dependent volumes, i.e., time dependent volumes that are not PVM coupling volumes, from the results of interpolations in user input time dependent tables.

2.40 Subroutine PMVSETJ

Subroutine PVMSETJ is similar to subroutine PVMSET except that it works on junction flow rate and velocity variables instead of volume variables.

2.41 Subroutine PVMSND

Subroutine PVMSND is the routine that sends messages to other tasks. It is divided into seven sections and the section to be executed is determined by the subroutine input parameter.

If the input parameter is zero, explicit parallel thermal-hydraulic messages are to be sent. The number of explicit messages to be sent is determined from the PVM internal file (file 31) and a loop over explicit send messages is executed. The loop first determines the whether the particular message is for parallel or sequential explicit thermal-hydraulic coupling by examining the explicit type flag. If the current message is for parallel explicit coupling (i.e., the explicit type flag for the message is negative one), the message tag and synchronous type for the message are determined from the PVM internal file. If the synchronous type specified by the calling routine matches the synchronous type of the message, the data to be sent are loaded into the local send buffer using the pointers into the RELAP5-3D© internal data files stored in the specifications for the message. Once all of the data are loaded into the local send buffer, the local send buffer is packed into the PVM send buffer using a call to the PVM library routine ‘pvmfpack’. The message is sent by a call to PVM library routine ‘pvmfsend’. Subroutine PVMSND then listens to receive the message acknowledgement by calling subroutine PVMFXREC. The processing of explicit messages continues after the message acknowledgement has been received until all explicit messages have been processed.

If the input parameter is one, a single semi-implicit thermal-hydraulic coupling messages is to be sent. The PVM internal file is examined to find the appropriate send message for this computational system and the message tag and task identifier of the receiving task are determined. The data are loaded into the RELAP5-3D© send buffer, the local send buffer is packed into the PVM send buffer, and the message is sent by a call to the PVM library routine ‘pvmfsend’. Subroutine PVMSND then listens to receive the message acknowledgement by a call to subroutine PVMFXREC. Subroutine PVMSND returns after the message acknowledgement has been received by subroutine PVMFXREC. There is no looping over messages because there can be only one send message per computational system for semi-implicit thermal-hydraulic coupling.

If the input parameter is two, a second type of semi-implicit thermal-hydraulic coupling message is to be sent. The procedure for sending the message is the same as the procedure for sending semi-implicit messages for a input parameter of one except that a different section of the PVM internal file is used to determine the message tags and sending task identifiers and to locate the data in the RELAP5-3D© internal files that is to be sent.

If the input parameter is three, a third type of message for semi-implicit thermal-hydraulic coupling message is indicated. The PVM internal file is examined to determine if the current computational system is to send semi-implicit coupling data. If the computational system is to send data, the name of the sending task is determined from the task identifier of the receiving task by a call to PVM library routine ‘pvmftasks’. If the name of the receiving task in not ‘relap5.x’, subroutine PVMSND returns. If the name of the sending task is ‘relap5.x’, the data is sent using the same procedure as for sending data when the input flag has a value of one or two. These messages are only sent by RELAP5-3D© if two instances of RELAP5-3D© are coupled to each other using semi-implicit coupling. These messages were included so that the results of coupling two instances of RELAP5-3D© to each other could be compared to the results of the simulation of the same problem using a single instance of RELAP5-3D© executed in uncoupled mode. The results of these two simulations were used to verify that the implementation of semi-implicit thermal-hydraulic coupling was working correctly.

If the input parameter is four, kinetics messages are to be sent. The PVM internal file is examined to determine the number of messages that are to be sent for kinetics coupling and a loop over messages is executed. The procedure for sending the messages is the same as for sending messages if the input parameter is zero, one, two, or three. The data are loaded into a local send buffer, the local send buffer is ‘packed’ into the PVM send buffer, and the message is sent. The subroutine then listens to receive the message acknowledgement, etc. This procedure is repeated until all kinetics messages have been sent.

If the input parameter in five, control system coupling messages are to be sent. The procedure for control system messages is the same as for kinetics messages except that a different portion of the PVM internal file is used to determine the message tag and the task identifier of the receiving task. The numerical identifier of the control component calling subroutine PVMSND is compared to the numerical identifier of the control component in the PVM internal file for the current message before loading the data into the local send buffer. The data are sent and the acknowledgement is received before proceeding to the next message in the section of the PVM internal file for control system send messages.

If the input parameter is six, messages containing volume data for the leader task of explicit sequential thermal-hydraulic coupling are to be sent. The message specification data for sequential explicit thermal-hydraulic messages are stored in the same portion of the PVM internal file as the specifications for messages for explicit parallel thermal-hydraulic coupling. The number of explicit messages is determined from the section of the PVM internal file for explicit messages and a loop over all explicit messages is executed. The explicit type for each message is examined and the message is sent if the explicit message type flag is one indicating that the message is for the follower task in explicit sequential thermal-hydraulic coupling. The data are loaded into the local send buffer if the synchronous type of the message matches the synchronous type specified by the calling routine (synchronous and asynchronous messages are sent by separate calls to subroutine PVMSND). The sending and receiving of all explicit message of whatever type is coordinated by the PVMEXEC program. Once the data are sent, subroutine PVMSND listens to receive an acknowledgment from the receiving task. The loop over explicit messages continues until all explicit send messages have been processed.

If the input parameter is seven, subroutine PVMSND performs different functions depending on whether it is called at the end of an asynchronous coupling interval, at the beginning of an asynchronous coupling interval, or whether it is called during an asynchronous coupling interval. Its overall responsibility is to compute the average flow rates in the coupling junctions during the coupling interval and to send the average flow rates to the coupled task at the end of the coupling interval. The subroutine is called at the end of each time step during an explicit synchronous coupling interval. If the subroutine is called at the beginning of an asynchronous coupling interval, it initializes the variables that will be used to compute the averages during the coupling interval. If it is called during the coupling interval, it increments the variables. If it is called at the end of the coupling interval, it increments the variables, computes the average values during the coupling interval, and sends the average values to the coupled task. The subroutine first loops over all explicit messages to find explicit messages whose explicit type has a value of one indicating that the message is from the leader process to the follower task in explicit sequential thermal-hydraulic coupling. Having found an appropriate message, it initializes, increments, and averages the junction flow rates as appropriate according to the control flag ‘exflag’. The procedure for actually sending the message is the same as the procedure for sending other explicit messages, the values to be sent are loaded into the send buffer, and the send buffer is sent to the coupled task by a call to PVM library ‘pvmfsend’. The subroutine then listens to receive the acknowledgement and returns after the acknowledgment is received.

2.42 Subroutine R3DCMP

Subroutine R3DCMP was modified to initialize variables related to PVM coupling in the database for three-dimensional components.

2.43 Subroutine RACCUM

Subroutine RACCUM was modified to initialize variables related to PVM coupling in the database for accumulator components.

2.44 Subroutine RBRNCH

Subroutine RBRNCH was modified to initialize variables related to PVM coupling in the database for branch components.

2.45 Subroutine RCONVR

Subroutine RCONVR was modified to initialize variables related to PVM coupling in the database for control components.

2.46 Subroutine RELAP5

Subroutine RELAP5 was modified to leave the virtual machine just before it terminates by calling PVM library ‘pvmfexit’ if RELAP5-3D$^{©}$ was executed under the control of the PVMEXEC program.

2.47 Subroutine RMTPLJ

Subroutine RMTPLJ was modified to initialize variables related to PVM coupling in the database for multiple junction components.

2.48 Subroutine RNEWP

Subroutine RNEWP was modified in several ways. At the start of a simulation run, it listens to receive a message with message tag 1 from the PVMEXEC program if RELAP5-3D$^{©}$ is being executed under the control of the PVMEXEC program. This message contains the start time, restart time, and simulation name that are to be used for the coupled simulation. It sends a message to the PVMEXEC program in a message with message tag 1 containing it’s restart status and then listens for a message from the PVMEXEC program that contains the global restart status. If the global restart was successful, it listens to receive a message with message tag 2 that contains data that is used to set the number of threads that RELAP5-3D$^{©}$ is to use for this coupled simulation. After the component data has been read, it calls the

PVM input processing routine RR5PVMC if a coupled simulation is being performed. It calls subroutine IR5PVMC after the first initialization routine ICMPN1 has been called and performs the initial data exchange for kinetics coupling sending power to the coupled task if this instance of RELAP5-3D© is the server task in kinetics coupling or receiving power data if this instance of RELAP5-3D© is the client task in kinetics coupling. After initializing the volumes and junctions, RELAP5-3D© initializes the heat structures using the initial power data read from it's input cards or the initial power data it has received from the coupled task. After initializing the heat structures, volumes and junctions, it listens to receive the thermal-hydraulic data from the coupled task (it's client task) that it needs for the initialization of the kinetics model if it is the server task in kinetics coupling. It then initializes the kinetics model. After the kinetics model has been initialized, the server task sends the revised power data to the client task and the client task listens to receive the initialized power data.

2.49 Subroutine RPIPE

Subroutine RPIPE was modified to initialize variables related to PVM coupling. in the database for pipe components.

2.50 Subroutine RPMVNJ

Subroutine RPMVNJ was modified to initialize variables related to PVM coupling in the database for pump components.

2.51 Subroutine RR5PVMC

Subroutine RR5PVMC is the subroutine that processes the messages received from the PVMEXEC program specifying the data that is to be exchanged with other tasks. It first determines the format for data transfers between itself and the PVMEXEC program using calls to PVM library routines to determine the type of architecture on which it is executing and the type of architecture being used by the PVMEXEC program. It the architectures are the same, the 'raw' mode of data transfers are used and if the architectures are dissimilar, 'xdr' mode is used.

After the data exchange format has been determined, storage space for internal file 31 is reserved in the RELAP5-3D© container array (i.e., the FA array). The storage space in file 31 is reserved and manipulated using calls to the 'ftb' routines. The arrangement of the data in file 31 is shown in Appendix C. The names of these library routines begin with the string 'ftb' and are part of the environmental subroutine library for RELAP5-3D©.

Next, a message with message tag 1000 is received from the PVMEXEC program. This message contains two data terms, the global wait time and the debug flag.

After the message with message tag 1000 is received, a message with message tag 1001 is received, This message contains the number of explicit thermal-hydraulic send and receive messages for this task. If the number of explicit send messages is greater than zero, a loop over the number of send messages is executed to receive messages with message tag 1002. The messages with message tag 1002 contain the specification (i.e., the metadata, or the data about the data) for the explicit send messages. The messages contain seven data items. The first data item is the messages tag to be used when sending the data. The second data item is the task identifier of the task to which the message is to be sent. The third item is an integer containing the flag denoting the synchronous type of the message (a value of zero denotes synchronous coupling and a value of one denotes asynchronous coupling). The fourth data item is an integer specifying the explicit type for the message (a value of -1 specifies parallel explicit coupling, a value of zero denotes the leader task in explicit sequential coupling, and a value of 1 denotes the follower task in explicit sequential thermal-hydraulic coupling). The fifth data item is a real number specifying the wait time for receiving the acknowledgement to the message. The sixth data item is an integer specifying the length of a string, and the seventh data item is the string variable describing the data items that are to be sent in the message.

After the acknowledgement to the message with message tag 1002 is sent to the PVMEXEC program, the number of blank delimited words in the string is determined by a call to subroutine NUMWORDS. If the number of words in the string returned by subroutine NUMWORDS is less than zero, The PVM error flag is set, the failure flag is set to true (stopping the computation at the end if input processing and initialization), and subroutine RR5PVMC returns to its calling routine. If the number of words returned by subroutine NUMWORDS is not an even number, an error condition is indicated, an error message is written to the output file, and the subroutine returns. If the number of blank delimited words in the string is an even number, the message specification has the correct format and the message specification is processed. The send tag, task identifier, the synchronous type, and the explicit type are written to the header portion of the metadata for the message. The metadata for a message contains two types of metadata, the metadata for the message as a whole contained in a header followed by the metadata for each variable that is to be sent in the message. The data format for the message is determined from the types of architecture for this task and from the task identifier of the task to which the messages is to be sent. After the data format is determined, it is written to the message header along with the wait time for the message. Global logical flags are set based on the explicit type and synchronous type of the message. Flag ‘pvmexc’ is true if either type of sequential coupling is specified in the message. Flag ‘pvmexca’ is true if explicit asynchronous sequential coupling is indicated by the values of the synchronous type and explicit type for the message. Flag ‘pvmexcs’ is true if the message is for explicit synchronous sequential coupling. The flags ‘pvmexa’ and ‘pvmexs’ are set to true for parallel explicit asynchronous coupling and for parallel explicit synchronous coupling, respectively. After the data has been stored in the header, the string is parsed into pairs of words for the determination of the data items that are to be sent in the message.

The first word in the pair is read by a call to subroutine GETWORD. This subroutine returns the value of the word in the position in the string that is specified in its input parameters. The word is compared to the names of the variables that can be plotted by RELAP5-3D© to determine if the word names a variable or a component. If the word names a variable, the name of the variable is loaded into the data portion of the metadata for the message and the second word of the pair is read and stored. If the first word in not the name of a plottable variable in RELAP5-3D©, the component database is searched for the matching component name. If the word does not match a component name, an error condition is indicated and the subroutine returns. If the word matches a component name, the component type is determined to find out whether the component is a volume type component or a junction type component. The volume components in RELAP5-3D© are pipe, time dependent volume, pump, branch, jetmixer, annulus, separator, single volume, accumulator, turbine, ECC mixer, multi-dimensional component, and pressurizer. The junction component types are multiple junction, time dependent junction, single junction, and valve. Depending upon the component type, the names of the variables to be sent are loaded from a list of variables defined in subroutine RR5PVMC. The volumes variables are pressure, vapor volume fraction, liquid volume fraction, specific internal energy for vapor, specific internal energy for liquid, mass quality of non condensable gas, density of vapor, density of liquid, temperature of vapor, temperature of liquid, and the mass equalities of five individual non condensable gases. The junction variables are junction vapor fraction, junction vapor density, junction vapor specific internal energy, junction mass quality of non condensable gas, junction vapor velocity, junction liquid volume fraction, junction liquid density, junction liquid specific internal energy, junction liquid velocity, and the junction mass qualities of the five individual non condensable gases. If the first word of a pair of words is a name of a variable, the second word of the pair is used as the numerical identifier of the variable. If the first word of a pair is a component name, the second word is the volume or junction number within the component. This number is converted to a numerical identifier by prepending the component number to the numerical value in the second word of the pair of words. The numerical identifier is entered into the metadata for each data item in the message. The variable name and the numerical identifier are now in a format that can be used by subroutine SCNREQ to locate the data items in the RELAP5-3D© database. The number of items in the message is incremented after each of the names is entered into the metadata for the message. Various other flags are set depending of the explicit type of the message.

After all of the messages with message tag 1002 have been processed, messages with message tag 1003 that specify the data to be received from other tasks for explicit thermal-hydraulic coupling are processed. The procedure for processing messages with message tag 1003 is the same as for messages with message tag 1002 except that the metadata for receive messages is stored after the metadata for send messages.

After all of the messages with message tags 1002 and 1003 have been processed, the metadata for explicit thermal-hydraulic messages is printed to the output file so that the user can determine if the messages contain the data that he intended to send and receive for explicit thermal-hydraulic coupling.

After the messages for explicit thermal-hydraulic coupling have been received from the PVMEXEC program, messages with message tags 1004, 1005, and 1006 are received from the PVMEXEC program for the specification of data for semi-implicit thermal-hydraulic coupling. Message 1004 contains the number of semi-implicit send and receive messages. Messages with message tag 1005 contain the specifications for send messages for semi-implicit coupling and messages with message tag 1006 contain the specifications for receive messages for semi-implicit coupling. After receiving message 1004 specifying the number of semi-implicit send and receive messages, subroutine RR5PVMC loops over the number of send and receive messages to receive messages with message tags 1005 and 1006 respectively.

Subroutine RR5PVMC divides messages with message tag 1005 into five data items. The first data item is the message tag for sending the message, the second data item is the task identifier of the task to which the data is to be sent, and the third data item is the wait time for the acknowledgement to the message. The fourth data item is the number of characters in a string variable, and the fifth item is a string variable of the specified length. The data format for the message is determined from the architectures of the CPUs on which this task and the coupled task are executing upon and the send tag, the task identifier for the task to which the message is to be sent, the data type, and the wait time are loaded into the metadata for the message. The string variable is checked for the correct format and the pairs of words are processed. The first word of a pair for semi-implicit coupling is a component name and the second word of a pair is the volume or junction number within the component. The component type is determined as explained above and the component number is converted into the RELAP5-3D$^{©}$ numerical identifier. The list of variables for volume components and junction components are listed in subroutine RR5PVMC. The lists are the same lists as for explicit thermal-hydraulic coupling. Several flags are set in the bit arrays for volumes and junctions to designate that the component is a coupling component.

The messages with message tag 1006 that specify the data that is to be received for semi-implicit coupling are processed in the same way as the send messages specified in 1005 messages. The data base for direct receive messages is different from the database for direct send messages in that storage locations are reserved in the PVM internal file for two copies of the data that is to be received in the messages. The data for direct semi-implicit receive messages in unpacked from the buffers in the PVM library routines into the ‘new’ storage locations instead of into the global receive buffer. Two copies of the values are saved, the ‘new’ values and the ‘old’ values. The ‘old’ values are used to restore the properties in the coupling components to their previous values if a time step failure has occurred and a backup is necessary.

After the send and receive message specifications have been processed, the metadata for semi-implicit coupling is read to generate another set of send and receive messages for semi-implicit coupling. The messages specified in the 1005 and 1006 messages from the PVMEXEC program are designated as the direct semi-implicit coupling messages because they are specified directly by the PVMEXEC program. The second set of messages for semi-implicit coupling are designated as the indirect semi-implicit messages because they are specified indirectly from the metadata for the direct messages. The metadata for the send messages are scanned to generate a new set of messages. These messages are sent to the same task as the corresponding direct message, but the message tag is incremented by a value of 2000 and the data specifications are taken from a different list built into subroutine RR5PVMC, the particular list depending on whether the direct message sends volume or junction data. The volume data are the pressure coefficients for the coupling volumes in the master task and the junction variables are the junction flow rates in the coupling junctions in the slave task. The corresponding indirect receive messages are generated from the metadata for the direct semi-implicit receive messages.

After the indirect semi-implicit send and receive messages are generated, a third set of semi-implicit messages are generated. These messages are designated as child messages. They are generated in the same way as the indirect messages except that the child send message is only generated if this instance of RELAP5-3D$^{©}$ is participating in semi-implicit coupling as the master task and the child receive message is only generated if this instance of RELAP5-3D$^©$ is participating in the semi-implicit coupling as the slave task. The messages are generated from the metadata for the direct send and receive messages except that their message tags are generated from the message tags of the direct messages by incrementing them by a value of 3000. The message tags for direct messages for semi-implicit coupling will lie in the range 1-999 because that is how they are assigned by the PVMEXEC program. The message tags in the range 2001-2999 and 3001-3999 are reserved for indirect and child messages for semi-implicit coupling.

The global semi-implicit coupling flag is set to designate that this instance of RELAP5-3D$^©$ is participating in semi-implicit coupling if the number of direct semi-implicit coupling messages is greater that zero. This completes the processing for the semi-implicit coupling.

Subroutine RR5PVMC next receives messages with message tags 1007, 1008, and 1009 for the specification of messages for kinetics coupling. The message with message tag 1007 specifies the number of send and receive messages for kinetics coupling. If the task is participating in kinetics coupling (i.e., the number of send and receive messages is greater than zero), it listens to receive a series of messages from the PVMEXEC code with message tag 1008. These messages contain the specifications for kinetics send messages. The messages with message tag 1008 contain five data items. The first data item is the message tag for the send message, followed by the task identifier of the task to which the message is to be sent. This is followed by the wait time for the acknowledgement to the message. The next data item is the number of characters in a string followed by a string variable containing the names and locations of the data items that are to be sent in the message. The send message tag, and the task identifier of the task to which the message is to be sent are stored in the message header. After the data format has been determined, the number of words in the string are checked and appropriate error messages are written if the format of the string is incorrect. If there are no errors in the string, it is parsed as pairs of words, the first word containing a keyword for the type of data that is to be sent or the name of a component and the second word specifying the locations in the system model from which the data is to be retrieved and sent to the coupled process. The kinetics coupling keywords and component names imply different lists of variables that are to be sent. In addition, a range of component numbers may be entered by separating the starting component number followed by a dash ('-') followed by the ending component number. The components in the range specified must be included in the system model and their component numbers must be in ascending numerical order with an increment of one. A flag is set designating that the task is the server task for kinetics coupling if the task is sending power variables to the coupled task.

After the kinetics send messages are processed, the kinetics receive messages are processed. The kinetics receive messages are specified in messages with message tag 1009 from the PVMEXEC program. The procedure for processing kinetics receive messages is the same as the procedure for processing kinetics send messages.

The last series of messages are for control systems coupling. The data for control systems coupling is contained in messages with message tags 1010, 1011, and 1012 with the message with message tag 1010 specifying the number of send and receive messages for control systems coupling. If the task is participating in control system coupling, it receives a series of messages with message tag 1011 from the

PVMEXEC program. These messages contain the specifications for control system send messages. Each message contains five data items. The first item is the message tag for the message, the second data item is the task identifier of the task to which the message is to be sent, and the third data item is the wait time for the acknowledgment to the message. The next data item is the number of characters in a string followed by the string. The string should contain pairs of items defining the data that is to be sent in the message. The string is examined for the proper format, and is parsed into pairs of words if there are no format errors. Each control system coupling control block may send any number of values to the receiving control block. The first pair of words specify the control component that is to send the data. Only the component number is significant and any meaningful name (i.e., the first word) may be used. The second data item specifies the control component identification number. A check for this control component should be added to the code and its control block type should be checked if the control component exists. After the data exchange format is determined from the architecture of the CPU on which this instance of RELAP5-3D is executing and the architecture of the CPU on which the receiving task is executing, the identifiers and component numbers of the data items that are to be sent are read from the string in pairs and loaded into the metadata for the message. The processing of pairs continues until there are no more pairs of data items in the string. Subroutine RR5PVMC continues receiving messages with message tag 1011 until the number of send messages specified in the message with message tag 1010 have been processed.

The processing for the control system receive messages is the same as for control system send messages. The receiving component (not the receiving control component but the component that is to receive the value) is checked to make sure that is an interactive variable and error messages and error flags are set appropriately.

Next, the solution algorithm selected by the user on his time step cards is checked if semi-implicit thermal-hydraulic coupling has been specified by the PVMEXEC program.

Finally, the size of internal file 31 is determined and the excess space reserved for file 31 is released. This completes the processing by subroutine RR5PVMC.

2.52 Subroutine RRESTF

Subroutine RRESTF was modified to read the name of the simulation contained in the header of the restart file and to compare the name read from the restart file to the name received from the PVMEXEC program for PVM coupled simulations. It sets the restart error parameter for the several different types of restart errors for PVM coupled problems.

2.53 Subroutine RRSTD

Subroutine RRSTD was modified to extend the length of the header for a non-restart simulation and writes the name of the simulation received for the PVMEXEC program to the restart file.

2.54 Subroutine RSNGJ

Subroutine RSNGJ was modified to initialize variables related to PVM coupling in the database for single junction components.

2.55 Subroutine RSNGV

Subroutine RSNGV was modified to initialize variables related to PVM coupling in the database for single volume components.

2.56 Subroutine RTMDJ

Subroutine RTMDJ was modified to initialize variables related to PVM coupling in the database for time dependent components.

2.57 Subroutine RTMDV

Subroutine RTMDV was modified to initialize variables related to PVM coupling in the database for time dependent volume components.

2.58 Subroutine RTSC

Subroutine RTSC was modified to set the start time of a coupled simulation to the value received from the PVMEXEC program.

2.59 Subroutine RTURB

Subroutine RTURB was modified to initialize variables related to PVM coupling in the database for turbine components.

2.60 Subroutine RVALVE

Subroutine RVALVE was modified to initialize variables related to PVM coupling in the database for valve components.

2.61 Subroutine SCNREQ

Subroutine SCNREQ was modified to find the locations (i.e., compute the value of a pointer) in the internal files (i.e., the FA array) of the several junction flow rate variables used by PVM coupling. It was also modified to find the location in the PVM internal file of the kinetics variables received from the coupled task in kinetics coupling. The variables in kinetics coupling messages contain variables for which no storage space has been reserved in the other internal files of the receiving task so they are stored in the PVM internal file. The variables in the messages for other types of coupling have storage space in the other internal files of the receiving task and can be saved in their default locations.

2.62 Subroutine SIMPLT

Subroutine SIMPLT was modified to compute the junction flow rate variables used by semi-implicit thermal-hydraulic coupling for all junctions in a simulation model so that they are available for plots and minor edits regardless of whether the simulation is coupled to another task on not coupled to another task.

2.63 Subroutine SRESTF

Subroutine SRESTF was modified to increase the length of the header of the restart file and to write the name of the simulation on the restart file for PVM coupled simulations.

2.64 Subroutine STATEP

Subroutine STATEP was modified to set the value of variable RHOM to the mixture density RHO for volumes that are PVM coupling volumes in the slave task in semi-implicit thermal-hydraulic coupling.

2.65 Subroutine STATEQ

Subroutine STATEQ was modified to set the value of variable RHOM to the mixture density RHO for volumes that are PVM coupling volumes in the slave task in semi-implicit thermal-hydraulic coupling.

2.66 Subroutine TRAN

Subroutine TRAN was modified to manage time step backups for coupled simulations. It also coordinates the data transfers for explicit sequential thermal-hydraulic coupling.

Subroutine TRAN was modified to compute the fluid properties in coupling junctions in the leader task for explicit sequential thermal-hydraulic coupling at the beginning of the first time step of an explicit coupling interval from the properties in the coupling volume that have been received from the follower task at the end of the previous coupling interval. After the junction properties are computed in the leader task, the task listens to receive the junction flow rates from the leader task. The leader task will terminate listening immediately and proceed with its advancements through the explicit coupling interval. The follower task will wait to receive the junction data from the leader task and then proceed with it's advancements once the junction data is received from the leader task. After the return from subroutine HYDRO (the subroutine that control and coordinates the thermal-hydraulic advancements), properties in the coupling junctions are computed in the leader task and are sent to the follower task if the task is participating in synchronous explicit sequential thermal-hydraulic coupling. After checking for synchronous explicit sequential coupling, subroutine TRAN checks for asynchronous explicit sequential coupling, computes the fluid properties in the coupling junctions, and sends the junction properties and flow rates to the follower task if the current time step is the last time step in an explicit coupling interval.

Imbedded between the sends and receives for explicit sequential thermal-hydraulic coupling, is the control logic for semi-implicit coupling. The logic for synchronous coupling was placed together because backups for failed advancements can occur for both explicit synchronous coupling and semi-implicit coupling (synchronous by default) but the exchange of junction data between the leader and follower tasks can only occur at the end of a successful advancement. Subroutine TRAN enters the backup logic if the task is participating in some type of synchronous coupling. The logic determines the type of backup from the global success flag and the flags for velocity flip-flop, water packing, and air appearance for the several computational systems. It determines a coupling success flag from these values and sends it to the PVMEXEC program in a message with message tag 10000. Subroutine TRAN than listens to receive the global success flag computed by the PVMEXEC program. If all synchronously coupled tasks have performed their advancement successfully, subroutine TRAN proceeds with the remainder of the computations for the advancement (i.e., the kinetics and control systems computations). If the coupling success flag indicates that the advancement should be repeated, the control logic sets the global success flag and the flags for the types of failure that caused the backup, and jumps to call subroutine DTSTEP to determine a time step size for the repeated advancement. Subroutine DTSTEP controls the computation of a new time step size and the data management for a repeated time step.

Once the thermal-hydraulic advancement is successful, the kinetics model is executed. Before the kinetics model is executed data are sent by the client task to the server task using calls to subroutine PVMSND, PVMRCV, and PVMPUT with input parameter 4. After the kinetics model has completed it’s computations, the kinetics power data is sent from the server task to the client task using the same set of calls to the PVM data transfer routines, PVMSND, PVMRCV, and PVMPUT.

After the data transfers for the kinetics model have finished, the control systems model is executed. The data transfers for the control systems model are controlled and performed by the control system coupling control block during the course of the advancement of the several control components that comprise the control systems model (i.e. subroutine CONVAR and subroutine CPLFNCTN).

This completes the description of the changes to subroutine TRAN for PVM coupling.

2.67 Subroutine TRNSET

Subroutine TRNSET has been modified for PVM coupling. First, the loop building the list of time dependent volumes in the slave task has been modified so that the coupling time dependent volume is added to the list regardless of the size of the table being used to specify its fluid conditions. This was done so that single entry table can be used for coupling volumes in the slave task. The same modification was made for time dependent junctions in the master task.

Next the pointers in the kinetics internal file (i.e., file 21) are set to point to the volume data in file 31 for the server task in kinetics coupling if appropriate. The pointers are converted from relative locations in the internal file into absolute locations in the FA array. The volume numbers for kinetics coupling volumes in the server task must be less that 1000000 to distinguish them from regular volumes in the server task. The pointers for heat structure data in the kinetics internal file are set the same way except that the heat structure number for coupling heat structures must be less that 10000 to distinguish them from regular heat structures in the server task. The modifications to pointers is done for both the point kinetics and the nodal kinetics pointers in the server task for kinetics coupling.

Subroutine TRNSET then reserves the space for the pointers to the pressure coefficients in the data file for each computational system. Next, the pointers for the data items in the RELAP-5D$^{©}$ third type of semi-implicit message are set if this instance of RELAP5-3D$^{©}$ is participating in semi-implicit coupling. This message will only be exchanged if two instances of RELAP5-3D$^{©}$ are coupled to each other using semi-implicit coupling. This completed the description of the modifications to subroutine TRNSET for PVM coupling.

2.68 Subroutine TSETSL

Subroutine TSETSL reserves the storage for the solution matrix and right hand sides for each computational system. It has been modified to reserve the storage for the extra right hand side vectors needed by the master task in semi-implicit coupling. It also sets the pointers to the pressure coefficients in the master task.

Subroutine TSETSL first bypasses the allocation of the matrix elements for the coupling junctions in the master task for semi-implicit coupling. These junctions will be treated as time dependent junctions whose flow rates are known (i.e., computed by the slave task). Next it reserves the space for the matrix elements for the coupling junction attached to the coupling volume in the slave task. The coupling volumes and coupling junctions in the slave task must be specified as time dependent volumes and time dependent junction is the system model in semi-implicit coupling and storage for their matrix elements is needed (the values for time dependent volumes and time dependent junctions is normally put in the right hand side vector for the system). After the storage for the matrix elements and right hand side vector has been reserved in the master task, the pointers in the indirect send message for the master task are set to point to the storage just reserved.

Finally, the pointers to the nonzero elements for the coupling junctions in the slave task are set in the loop storage arrays for the coupling junctions in the slave task so that the contributions of the pressure coefficients from the master task might be added to the system matrix in the slave task.

The author of this report is well aware that the explanation of the storage and pointers for the solution matrix is not very comprehensive. The difficulty is that there is no explanation of how the matrix elements and right hand side vectors are stored for uncoupled problems to motivate the discussion of the changes to how the matrix elements and right hand side vectors are stored.

2.69 Subroutine TSTATE

Subroutine TSTATE was modified to call subroutines PVMSET and PVMSETJ that facilitate the computation of the fluid properties and junction velocities for PVM coupling components.

2.70 Subroutine VEXPLT

Subroutine VEXPLT was modified to reset the predicted velocities and velocity derivatives on the second attempted advancement for the time step for junctions where water packing (or fluid stretching) occurs if semi-implicit thermal-hydraulic coupling is being used.

2.71 Subroutine VFINL

Subroutine VFINL has several modifications for PVM coupling. The number of right hand side vector is incremented by seven times the number of PVM coupling junctions if this instance of RELAP5-3D© is serving as the slave task in semi-implicit thermal-hydraulic coupling and the default BPLU solver is being used. The several junction flow rate variables defined for semi-implicit coupling are computed for junctions that are not time dependent junctions and are not PVM coupling junctions in the master task in semi-implicit thermal-hydraulic coupling. Next comes several modifications for the time step failure and backup logic for time step failures that cause a partial time step repeat with the same time step size for uncoupled simulations. These failures cause a full time step repeat for synchronously coupled simulations because the coupled task needs to repeat the attempt as well as the task that experiences the failure. First, the solution is checked for velocity flip-flop and the velocity flip-flop flag is set and the advancement is repeated if velocity flip-flop has occurred. On the repeat for velocity flip-flop, or if velocity flip flop has not occurred on the first attempt for an advancement, the solution is checked for water packing, the water packing flag is set, and the advancement repeated if water packing has occurred. On a repeat for water packing, or if water packing has not occurred in the first attempt, the solution is checked for the appearance of non condensable gases. If non condensable gas has appeared in a volume that has no non condensable gas at the beginning of the advancement, the non condensable gas appearance flag is set, and the advancement is repeated. The failure flags are checked so that the velocity flip-flop check is bypassed if the current attempted advancement was caused by velocity flip-flop in the previous attempted advancement, etc. However, the failures may occur in any order, thus a velocity flip-flop may occur on the repeat caused by a water packing failure, etc. This logic is so complicated that is it not clear whether the coupled codes might not get into a deadlock in which one task fails because of velocity flip-flop, the repeat causes a water pack in the other coupled task, and that the repeat for the water pack might cause an different failure in the task that first caused the initial repeat, i.e., the failures are caused by alternate tasks on alternate repeated attempts for the advancement and the repeats cause an infinite loop. The correctness of the backup logic has not been investigated thoroughly.

2.72 Subroutine VLVELA

Subroutine VLVELA has been modified to compute the volume velocity in PVM coupling volumes in the follower task of explicit sequential thermal-hydraulic coupling at the beginning of the time step advancement. The volume velocity is computed for all volumes at the end of the advancement.

2.73 Subroutine WPIDDA

Subroutine WPIDDA has been modified to write the labels for the several junction flow rates used by PVM coupling to the direct access plot file.

2.74 Subroutine WRPLID

Subroutine WRPLID has been modified to write the labels for the several junction flow rates used by PVM coupling to the sequential access plot file.

2.75 Comdeck CNVTPA

Comdeck CNVTPA was modified to increase the value of the parameter defining the length of the array containing the names of the types of control components.

2.76 Comdeck CNVTPAD

Comdeck CNVTPAD was modified to add the name of the control component used for PVM coupling on the control systems model.

2.77 Comdeck JUNDAT

Comdeck JUNDAT was modified to reserve storage for the new junction flow rate variables used by PVM coupling.

2.78 Comdeck JUNDATC

Comdeck JUNDATC was modified to add descriptions of the new junction flow rate variables added to Comdeck JUNDAT.

2.79 Comdeck LPDAT

Comdeck LPDAT was modified to reserve storage for several new variables used by PVM coupling.

2.80 Comdeck LPDATC

Comdeck LPDATC was modified to add descriptions of the new PVM coupling variables added to Comdeck LPDAT.

2.81 Comdeck LVECTR

Comdeck LVECTR was modified to reserve storage for a new pointer for PVM coupling.

2.82 Comdeck PVMVR5

Comdeck PVMVR5 was added to the RELAP5-3D© code for the original implementation of explicit PVM coupling. It was modified extensively to add the new types of coupling. Appendix A contains the definitions of the parameters and variables in this comdeck.

2.83 Comdeck PVMVR5C

Comdeck PVMVR5C contains descriptions of the PVM coupling variables in comdeck PVMVR5.

2.84 Comdeck R5PVMCP

Comdeck R5PVMCP was added to the RELAP5-3D© code for the original implementation of explicit PVM coupling. It was modified extensively to add the new types of coupling. Appendix A contains the definitions of the parameters and variables in this comdeck

2.85 Comdeck R5PVMCPC

Comdeck R5PVMCPC contains descriptions of the parameters and variables in comdeck R5PVMCP.

2.86 Comdeck VOLDAT

Comdeck VOLDAT was modified to reserve storage for a new PVM coupling variable for the volumes.

2.87 Comdeck VOLDATC

Comdeck VOLDATC was modified to add a description of the new PVM coupling variable for volumes.

2.88 Comdeck VREQD

Comdeck VREQD was modified to add the plot request names of the several junction flow rate variables used by PVM coupling.

2.89 Comdeck VREQS

Comdeck VREQS was modified to reserve storage for the names of the several junction flow rate variables used by PVM coupling.

3 References

1 W. L. Weaver, *The Application Programming Interface for the PVMEXEC Program and Associated Code Coupling System*, INL/EXT-05-00107, Idaho National Laboratory, March 2005.

2 W. L. Weaver, *Programmers Manual for the PVMEXEC Program*, INL/EXT-05-00159, Idaho National Laboratory, March, 2005.

3 C. Y. Paik and L. E. Hochreiter, *Analysis of FLECHT SEASET 163-Rod Blocked Bundle Data Using COBRA-TF*, NUREG/CR-4166, US Nuclear Regulatory Commission, 1986.

4 The RELAP5 Code Development Team, *RELAP5-3D Code Manuals, Vol I, II, IV, and V,* Idaho National Engineering and Environmental Laboratory, INEEL-EXT-98-00834, Revision 2.2, October 2003.

5 Fluent Corporation, *Fluent 6 User's Guide,* 2001.

6 Safety Code Development Group, *TRAC-PF1/MOD1: An Advanced Best-Estimate Computer Program for Pressurizer Water Reactor Thermal-Hydraulic Analysis,* LA-10157-MS, NUREG/CR-3858, Los Alamos National Laboratory, July 1986.

7 J. W. Spore, et. al., *TRAC-M/FORTRAN 90 (Version 3.0) Theory Manual,* NUREG/CR-6724, Los Alamos National Laboratory and Pennsylvania State University, July 2001.

8 K. K. Murata et. al., *Code Manual for CONTAIN 2.0: A Computer Code for Nuclear Reactor Containment Analysis,* SAND97-1735, NUREG/CR-6533, Sandia National Laboratory, December 1997.

9 R. O. Gauntt et. al., *MELCOR Computer Code Manuals: Version 1.8.5,* SAND2000-2417, NUREG/CR-6119, Sandia National Laboratory, October 2000.

10 P. J. Turinsky, R. M. K. Al-Chalabi, P. Engrand, *NESTLE: A Few-Group Neutron Diffusion Equation Solver Utilizing the Nodal Expansion Method for Eigenvalue, Adjoint, Fixed Source Steady State and Transient Problems,* EGG-NRE-11406, Idaho National Engineering Laboratory, 1994.

11 T. Downar et. al., *PARCS: Purdue Advanced Reactor Core Simulator, PU/NE-98-26,* Purdue University, West Lafayette, IN, September, 1998.

12 A. Geist et. al., *PVM (Parallel Virtual Machine) User's Guide and Reference Manual,* Oak Ridge National Laboratory, ORNL/TM-12187, 1993.

13 R. P. Martin, "RELAP5/MOD3 Code Coupling Model," *Nuclear Safety*, *Vol. 36, No. 2*, pp. 290-299, July-December 1995.

Appendix A
Data Dictionary for PVM Coupling Variables

This appendix lists the variables added to existing comdecks in RELAP5-3D© for the PVM coupling. The variables in new comdecks added for PVM coupling are also listed. The variables are listed by comdeck and the comdecks are listed in alphabetical order.

The data items added to comdeck JUNDAT are:

Variable	Definition
uflowgj	Flow rate of internal energy in the vapor field
uflowfj	Flow rate of internal energy in the liquid field
aflowgj	Flow rate of noncondensable gas
vflowgj	Volumetric flow rate of vapor
vflowfj	Volumentric flow rate of liquid
flentha	Flow rate of enthalpy in noncondensable gas
flenthg	FLow rate of enthalpy in vapor
flenthf	Flow rate of enthalpy in liquid

The data items added to comdeck LPDAT are:

Variable	Definition
cpltid	Task identifier of task coupled to this task using semi-implicit coupling for this conputational system
nvpvmc	Number of PVM coupling volumes in this computational system
slvprc	Maste/slave flag for semi-implicit coupling of this computational system

The data items added to comdeck LVECTR are:

Variable	Definition

lvcofp	Pointer to PVM pressure coefficients for this computational system if master task in semi-implicitly coupled

The data items in comdeck PVMVR5 are:

Variable	Description
recvar	Data array for receive buffer
sndvar	Data array for send buffer
wait	Global wait time
pvmdt	Time step for synchronously coupled simulation
start_time	Start time of the coupled simulation
restart_time	Restart time for the restart of a coupled simulation
excplint	Duration of the previous explicit coupling interval
curtex	Time for the next explicit coupling data exchange
pvmname	Name of the coupled simulation
rstname	Name of simulation on restart file
pvmexs	Control flag. True if task is participating in some type of explicit synchronous coupling
pvmexa	Control flag. True if task is participating in some type of explicit asynchronous coupling
pvmsyn	Control flag. True if task is participating is some form of synchronous coupling
pvmexc	Control flag. True is task is participating is some form of sequentail explicit coupling.
pvmexca	Control flag. Ture if task is participating in explicit sequential asynchronous coupling
pvmexcs	Control flag. Ture is task is participating is explicit sequential synchronous coupling

cplfnctn	Component number of control component sending or receiving control system coupling messages
rkserv	Control flag. Value is zero if task is client in kinetics coupling and value is one if task is server in kinetics coupling.
pvmdbg	Debug flag. Received from PVMEXEC program. Can be used to activate debug printout, etc.
mytid	Task identifier of task
extid	Task identifier of PVMEXEC program
pvmerr	Error flag for coupling
sndtid	Task identifier of sending task for explicit coupling. Specified by PVMEXEC program
sync	Control flag for explicit coupling data exchanges. Determines whether synchronous or asynchronous messages are being exchanged.
pvm_success	Success flag for synchronously coupled simulations.
exdatatype	Data format flag for messages with the PVMEXEC program
renode_pvm	Not used
exflag	Control flag for explicit coupling data exchanges.
jflag	Control flag for computation of junction properties for explicit sequential coupling.
vflag	Control flag for computation of volume velocities for explicit sequential coupling.
pvm_backup	Control flag in semi-implicit coupling for data transfers during a backup and for data storage for a successful advancement.

The following parameters are defined in comdeck PVMVR5:

Parameter	Definition
nhdexp	Number of data items in header for explicit messages

nhdimp	Number of data items in header for semi-implicit messages
nhdkin	Number of data items in header of kinetics messages
nhdctl	Number of data items in header for control system compling messages
nibexp	Number of data items in metadata for each variable in explicit coupling messages
nibimp	Number of data items in metadata for each variable in semi-implicit coupling messages
nibkin	Number of data items in metadata for each variable in kinetics coupling messages
nibctl	Number of data items in metadata for each variable in control systel coupling messages

The data items in comdeck R5PVMCP are:

Variable	Definition
nr5ccp	Integer variable for fixed portion of file 31
ni5snd	Integer variable for data items in send messages
ni5rcv	Integer variable for data items in receive messages
nr5snd	Real variable for data items in send messages
nr5rcv	Real variable for data items in receive messages
na5snd	Character variable for data items in send messages
na5rcv	Character variable for data items in receive messages
pvwait	Global wait time (same as variable wait in comdeck PVMVR5)

The following parameters are define in comdeck R5PVMCP:

Variable	Definition
nvpitm	Number of items in list of volume variables for direct messages in semi-implicit coupling

njpitm	Number of items in list of junction variables for direct messages in semi-implicit coupling
nvcitm	Number of items in list of volume variables for slave messages in semi-implicit coupling
njfitm	Number of junction flow rates in list of junction variables for indirect messages in semi-implicit coupling
njsitm	Number of junctions flow rates in list of junction variables for indirect messages in semi-implicit coupling used in semi-implicit coupling solution algorithm
nvcitm3	Number of three-dimensional variables in list of volume variables for slave messages in semi-implicit coupling
nvkitm	Number of varaibles in list of volume variables for kinetics coupling
nznitm	Number of variables in list of zone variables for kinetics coupling
npwitm	Number of variables in list of power variables in kinetics coupling
nvklen	Length of storage block for volume variables in server task in kinetics coupling

The variables added to comdeck VREQD are as follows:

Variable	Definition
uflowgj	Junction flow rate of internal energy in vapor
uflowfj	Junction flow rate of internal energy in liquid
aflowgj	Junction flow rate of noncondensable gas
vflowgj	Volumetric flow rate of vapor in junction
vflowfj	Volumetric flow rate of liquid in junction

Appendix B Error Codes

This appendix lists the error codes that are loaded into the pvmerr variable. The values are listed along with a short description of the error condition.

Value	Description
1	A task has requested a time step size less than the minimum time step size.
2	A task cannot restart at the requested restart time.
4	A task cannot perform a restart because the simulation name on its restart file is not the same as the simulation name for this coupled run.
5	The time for the final restart on the restart file for a task is not the same time as the final time on the restart files of the other tasks.
6	The correct time step interval cannot be found on the time step cards.
10002	A task has time out waiting for a message from another simulation task.
10001	A simulation task has failed.
-mytid	The executive task has timed out waiting for a message from a simulation task where mytid is the task identifier of the PVMEXEC program.

Appendix C
Data Layout in RELAP5-3D ‘File’ 31

This appendix describes the arrangement of the data in the dynamic storage ‘file’ 31. This ‘file’ contains the metadata for the PVM coupling methodology. The first section of the data file contains 22 data items. The data items are as follows:

Data Item	Description
1. nr5ccp	Number of data exchange frequency pairs (obsolete, not used)
2. nr5ccp	Pointer to explicit send message metadata
3. nr5ccp	Pointer to explicit receive message metadata
4. nr5ccp	Number of explicit send messages
5. nr5ccp	Number of explicit receive messages
6. nr5ccp	Pointer to direct semi-implicit send message metadata
7. nr5ccp	Pointer to direct semi-implicit receive message metadata
8. nr5ccp	Number of direct semi-implicit send messages
9. nr5ccp	Number of direct semi-implicit receive massages
10. nr5ccp	Pointer to indirect semi-implicit send message metadata
11. nr5ccp	Pointer to indirect semi-implicit receive message metadata
12. nr5ccp	Pointer to slave semi-implicit send message metadata
13. nr5ccp	Pointer to slave semi-implicit receive message metadata
14. nr5ccp	Number of kinetics coupling send messages
15. nr5ccp	Number of kinetics coupling receive messages
16. nr5ccp	Pointer to kinetics send message metadata
17. nr5ccp	Pointer to kinetics receive message metadata
18. nr5ccp	Number of control system send messages
19. nr5ccp	Number of control system receive messages
20. nr5ccp	Pointer to control system coupling send message metadata
21. nr5ccp	Point to control system coupling receive message metadata
22. pvwait	Wait time

The next section contains the metadata defining the send data for explicit coupling messages. This section is broken down into subsections, one subsection for each send message. The data for a subsection consists of a header of seven data items followed by blocks of six data items, one block of six data items for each variable in the message. The first data item in the header is the message identifier for the message. The second data item in the header defines the number of sextuples of data in the subsection. The third data item is the PVM task identifier of the process to which the data is to be sent. The fourth data item in the message header denotes a synchronous or asynchronous message. The fifth data item defines the type of explicit coupling. The sixth data item is the data format flag for the message. The last data item in the message header is the wait time for the message acknowledgment. The message header is followed by sextuples of data items that describe the data in the message. The first word of the sextuple is a character string containing the name of the variable, i.e. the name of the variable in a plot request. The second word of a sextuple is an integer describing the location of the data item, i.e. the plot request parameter consisting of a packed integer containing the component identification number and the location in the component. The third word of the sextuple is an integer containing the file number of the internal data file, i.e., the 'FA' file number, containing the data item. The fourth word of the sextuple contains the relative location in the internal file of the data item. The fifth word of the sextuple is an integer containing the averaging flag for explicit sequential coupling. The last data items is a real number containg the average value of the variable. The block of six data items is repeated for each data item in the message. The metadata for the second explicit send message follows the metadata for the first explicit send message, etc. This data is located by using the pointer contained in variable nr5ccp(2).

1. ni5snd	Explicit send message tag
2. ni5snd	Number of items in send message
3. ni5snd	Task identifier of task receiving data
4. ni5snd	Synchronous flag (0 = synchronous, 1 = aysnchronous)
5. ni5snd	Explicit type of send message (-1 = parallel, 0 = leader, 1 = follower)
6. ni5snd	Data format flag.(0 = xdr, 1 = raw)
7. nr5snd	Wait time for acknowledgment
8. na5snd	Variable name of first item in send message
9. ni5snd	Volume or junction number of data item
10. ni5snd	File index of 'file' containing data item
11. ni5snd	Offset in 'file' to data item
12. ni5snd	Averaging flag for variable (used in sequential explicit coupling
13. nr5snd	Average value of variable (used in sequential explicit coupling)
14. na5snd	Variable name of second item in send message
15. ni5snd	Volume or junction number of second data item

....
....
....

The next section is similar to the previous section except that it specifies the metadata for data to be received by the process if the explicit coupling methodology is being utilized. The layout of the data for explicit receive messages is the same as for explicit send messages. This section of data is located by using a pointer contained in nr5ccp(3).

1. ni5rcv	Explicit receive message tag
2. ni5rcv	Number of items in receive message
3. ni5rcv	Task identifier of task sending data
4. ni5rcv	Synchronous flag
5. ni5rcv	Explicit type of message
6. ni5rcv	Data format for acknowledgment
7. ni5rcv	Wait time for message
8. na5rcv	Variable name of first data item in receive message
9. ni5rcv	Volume or junction number of data item
10. ni5rcv	File index of 'file' containing data item
11. ni5rcv	Offset in 'file' to data item
12. ni5rcv	Not used
13. nr5rcv	Not used
14. na5rcv	Variable name of second data item in receive message
15. ni5rcv	Volume or junction number of second data item

....
....
....

The next section of the data file contains metadata specifying the data in messages for the semi-implicit coupling methodology. The messages are called 'direct' because the contents of the messages are specified on the input cards. If a volume is named on an input card, the message sends the conditions in the volume to the coupled code. If a junction is named on the input cards, the junction conditions are sent to the coupled code.

The indirect messages in the following sections are implied by the volumes or junctions named in direct messages. If a volume is named on the input cards, by implication, the volume pressure coefficients for that volume must be sent to the other process. If a junction is named on the input cards, by implication, the phasic mass flow rates for that junction must be sent to the coupled process. Also by implication, if volumes are named on the input cards, the volume centered velocities in these volumes must be sent to the coupled code in the slave send messages if RELAP5-3D to RELAP5-3D coupling is specified.

The message tag numbers for the implied messages are the values of the corresponding direct messages plus 2000 for the indirect messages and the values of the corresponding direct messages plus 3000 for the slave messages. The data for each send message consists of seven data items followed by blocks of quintuplets of data. The number of quintuplets of items is determined by whether the items are volume items or junction items and whether the message is a direct, indirect, or a slave message. The direct semi-implicit send messages are located by the pointer contained in nr5ccp(6). The direct messages for volumes contain the pressure, void fraction, vapor specific internal energy, liquid specific internal energy, non-condensable gas mass fraction, vapor density, and liquid density for each volume.The direct messages for junctions contain the donor void fraction, the donor vapor density, the donor vapor specific internal energy, the donor non-condensable gas mass fraction, the vapor velocity, the donor liquid fraction, the donor liquid density, the donor liquid specific internal energy, and the liquid velocity for each junction. The quintuples of data items describe the data in the send and receive messages. The first word of a quintuple is a character variable that contains the name of the component containing the data item. The second word of a quintuple is a character variable containing the name of the variable as specified in a plot request. The third word is an integer specifying the location in the component as specified in a plot request. The fourth word is an integer specifying the internal file containing the data, e.g., a value of four for data items in volumes, etc. The fifth data item is an integer specifying the relative location in the file where the data item is found. Each data item in a send or receive message is described by a quintuple.

1. ni5snd	Direct semi-implicit send message tag
2. ni5snd	Index of system containing data items
3. ni5snd	Number of items in message
4. ni5snd	Task identifierof task receiving data
5. ni5snd	Master/slave flag(0 = master, 1 = slave)
6. ni5snd	Data format flag
7. nr5snd	Wait time for acknowledgment
8. na5snd	Name of component containing first item
9. na5snd	Variable name of first data item
10. ni5snd	Volume or junction number of first data item
11. ni5snd	File index of ‘file’ containing first data item
12. ni5snd	Offset in ‘file’ containing first data item
13. na5snd	Name of component containing second data item
14. na5snd	Variable name of second data item
15. ni5snd	Volume or junction number of second data item

....
....
....

The next section of the data file contains the metadata for the direct receive messages. It is arranged identically to the send message section and is located using the pointer contained in variable nr5ccp(7). The metadata describing the variables in an indirect receive message is followed by storage space for two copies of the data contained in the message. These storage locations are used to hold the current values of the variables received from the coupled task and the previous values of the same variables, i.e., the 'new' or end of time step values of the variables and the 'old' or beginning of time step values of the variables. The 'new' values are written over the 'old' values at the end of a successful advancement and the 'old' values are written over the 'new' values on a backup for a failed advancement, just like the values in the volume and junction internal files. The direct receive messages contain the same data items for volumes and junctions as the direct send messages, i.e., pressure, void fraction etc. for volumes and donor void fraction, donor vapor density, etc. for junctions.

1. ni5rcv	Direct semi-implicit receive message tag
2. ni5rcv	Index of system containing data items
3. ni5rcv	Number of items in message
4. ni5rcv	Task identifier of tasksending data
5. ni5rcv	Master/slave flag
6. ni5rcv	Data format flag for acknowledgment
7. nr5rcv	Wait time for message
8. na5rcv	Name of component containing first item
9. na5rcv	Variable name of first data item
10. ni5rcv	Volume or junction number of first data item
11. ni5rcv	File index of 'file' containing first data item
12. ni5rcv	Offset in 'file' containing first data item.
13. na5rcv	Name of component containing second data item
14. na5rcv	Variable name of second data item
15. ni5rcv	Volume or junction number of second data item
....	
....	
....	
16. nr5rcv	First 'new' variable
17. nr5rcv	Second 'new' variable
....	
....	
18. nr5rcv	Last 'new' variable
19. nr5rcv	First 'old' variable
20. nr5rcv	Second 'old' variable
21.	

22.	
23. nr5rcv	Last ‘old’ variable
24. ni5rcv	Message tag for second direct semi-implicit receive message
25. ni5rcv	Index of system receiveing second direct semi-implicit receive message

....

The next section contains the specification for the indirect semi-implicit send messages. The data for each message consists of six data items followed by a number of subsections. Each subsection consists of data for a single coupling volume. The data in each subsection consists of a quintuplet of data items followed by seven quintuplets of data for each coupling junction in the computational system. The data are located in the file using the pointer contained in variable nr5ccp(10). The data items for each volume consist of a constant (first quintuplet) followed by the seven pressure coefficients (seven quintuplets) for each coupling junction. The pressure coefficients are the change in the pressure with respect to the mass flow rate of non-condensable gas, the change in pressure with respect to the flow rate of vapor internal energy, the change in the pressure with respect to the flow rate of liquid internal energy, the change in pressure with respect to the flow rate of vapor mass, the change in pressure with respect to the flow rate of liquid mass, the change in pressure with respect to the volumetric flow rate of vapor, and the change in pressure with respect to the volumetric flow rate of liquid. The data items for each junction in an indirect semi-implicit send message are the flow rate of non-condensable gas, the flow rate of vapor internal energy, the flow rate of liquid internal energy, the mass flow rate of vapor, the mass flow rate of liquid, the volumetric flow rate of vapor, and the volumetric flow rate of liquid.

1. ni5snd	Indirect semi-implicit send message tag (value is 2000 plus the value of the corresponding direct message tag)
2. ni5snd	Index of computational system containing data items
3. ni5snd	Number of data items in message
4. ni5snd	Task identifier of task receiving data
5. ni5snd	Master/slave flag
6. ni5snd	Data format flag
7. nr5snd	Wait time for acknowledgment
8. na5snd	Name of component containing first data item
9. na5snd	Variable name of first data item in message
10. ni5snd	Volume or junction number for first data item
11. ni5snd	File index of ‘file’ containing first data item
12. ni5snd	Offset in ‘file’ for first data item
13. na5snd	Name of component containing second data item
14. na5snd	Variable name for second data item

....
....
....

The next section contains the specification for the indirect receive messages. The layout of the data is the same as for the indirect send messages. The data are located in the file using the pointer contained in the variable nr5ccp(11).

1. ni5rcv	Indirect semi-implicit receive message tag (value is 2000 plus value of corresponding receive message tag)
2. ni5rcv	Index of computational system containing data items
3. ni5rcv	Number of data items in message
4. ni5rcv	Task identifierof task sending data
5. ni5rcv	Master/slave flag
6. ni5rcv	Data format flag
7. nr5rcv	Wait time for message
8. na5rcv	Name of component containing first data item
9. na5rcv	Variable name of first data item in message
10. ni5rcv	Volume or junction number for first data item
11. ni5rcv	File index of ‘file’ containing first data item
12. ni5rcv	Offset in ‘file’ for first data item
13. na5rcv	Name of component containing second data item
14. na5rcv	Variable name for second data item

....
....
....

The next two sections contain the specification for the slave semi-implicit send and receive messages. The layout of these sections is identical to the semi-implicit direct and indirect receive messages. The slave semi-implicit send and receive messages can be located by use of the pointers stored in the variables nr5ccp(12) and nr5ccp(13) respectively. The data items in the slave messages consist of the volume centered velocities for each phase, the total heat flux from all heat structure surfaces, the total heat flux from heat structure surfaces to vapor, the volume centered velocities for the wall friction computation for each phase, and the donored velocities for each phase followed by the near wall boiling and condensation vapor generation rates. Phasic velocities are sent for each of the three coordinate directions for a total of twenty two data items per coupling volume. The twenty two items for a volume is followed by the surface temperatures of any heat structures connected to the volume. The layout of the receive data is identical to the layout of the send data and is not repeated.

1. ni5snd	Slave semi-implicit send message tag (value is 3000 plus the value of the corresponding direct message tag)

2. ni5snd	Index of computational system containing data items
3. ni5snd	Number of data items in message
4. ni5snd	Task identifier of task receiving data
5. ni5snd	Master/slave flag
6. ni5snd	Data format flag
7. nr5snd	Wait time for acknowledgment
8. na5snd	Name of component containing first data item
9. na5snd	Variable name of first data item in message
10. ni5snd	Volume or junction number for first data item
11. ni5snd	File index of 'file' containing first data item
12. ni5snd	Offset in 'file' for first data item
13. na5snd	Name of component containing second data item
14. na5snd	Variable name for second data item

....
....
....

The next two sections contain the metadata for the send and receive messages for kinetics coupling. In addition to containing the specification for the data in the receive messages, the receive data area also contains space to store the received data. The kinetics send and receive data areas can be located through pointers in nr5ccp(16) and nr5ccp(17), respectively. The send data consists of volume and heat structure data for clients and power data for the server process. Receive data consists of volume or heat structure data for the server process and power data for the client processes. Legal component names are the names of volume components and the keywords 'zone', 'heatstr, and 'power'.The layout of the send data is as follows: each message begins with a header of six items followed by blocks of five items, one block for each data item. Volume data consists of eight items, heat structure data of one item, power data of five items and zone data of five items.

1. ni5snd	Message tag for kinetics send message
2. ni5snd	Number of data items in message
3. ni5snd	Task identifier of task receiving message
4. ni5snd	Not used
5. ni5snd	Data format flag
6. nr5snd	Wait time for acknowledgment
7. na5snd	Name of component containing first data item
8. na5snd	Variable name of first data item
9. ni5snd	Component number of first data item

10. ni5snd	File index of ‘file’ containing variable
11. ni5snd	Offset in ‘file’ for first data item
12. ni5snd	Name of component containing second data item
13. ni5snd	Variable name of second data item

....

The layout of the kinetics receive data is similar to that of the kinetics send data except that extra storage is reserved to store the data received in each message. The number of extra storage locations reserved for the data depends on the type of data received. Five extra storage locations are reserved for power and zone data, one extra storage location is reserved for heat structure data and seventeen extra storage locations are reserved for volume data. The extra storage is located immediately after the blocks that define the data. Legal data type names are the keywords ‘volume’, ‘heatstr’, ‘zone’, and ‘power’. The layout is as follows.

1. ni5rcv	Message tag for first receive message
2. ni5rcv	Number of data items in first receive message
3. ni5rcv	Task identifier of task receiving this message
4. ni5rcv	Currently not used
5. ni5rcv	Data format flag for acknowledgment
6. nr5rcv	Wait time for message
7. na5rcv	Data type name for first data item (e.g., power)
8. na5rcv	Variable name of first data item (e.g., rkpow)
9. ni5rcv	Data identification number of first data item (e.g., 0)
10. ni5rcv	File index of ‘file’ containing first data item
11. ni5rcv	Offset in ‘file’ for first data item
12. na5rcv	Data type name of second data item
13. na5rcv	Variable name of second data item
14. ni5rcv	Data identification number of second data item
15. ni5rcv	File index of ‘file’ containing second data item
16. ni5rcv	Offset in ‘file’ for second data item
17.	
18. na5rcv	Data type name for ‘last’ data item in message
19. na5rcv	Variable name of ‘last’ data item in message
20. ni5rcv	Data type identification number of ‘last’ data item
21. ni5rcv	File index of ‘file’ containing last data item
22. ni5rcv	Offset in ‘file’ for last data items

23. nr5rcv	First storage location for values in receive message
24. nr5rcv	Second storage location for values in receive message
25.	
26. na5rcv	Data type name for second data type specification in message
27. na5rcv	Variable name for first data items in second data type specification
28. na5rcv	Data type identifier for first data items in second data type specification

......

The last section of the PVM data file contains the control system coupling metadata. The data areas for the send and receive data are located through pointers in nr5ccp(20) and nr5ccp(21), respectively. The send data consists of data items that can be used in minor edits or for plotting. Receive data is stored in user defined interactive variables. The layout of the send data is as follows: each message begins with a header of 5 items followed by blocks of 4 items, one block for each data item.

1. ni5snd	Message tag for control system send message
2. ni5snd	Number of data items in message
3. ni5snd	Task identifierof task receiving message
4. ni5snd	Control block number of 'extfnctn' control block sending message
5. ni5snd	Data format flag
6. nr5snd	Wait time for acknowledgment
7. na5snd	Name of first send variable
8. ni5snd	Component number of first send variable
9. ni5snd	File index of 'file' containing data item
10. ni5snd	Offset in 'file' for first data item
11. na5snd	Name of second send variable
12. ni5snd	Component number of second send varaible

....

The layout of the control system receive metadata is similar to the layout of the control system send metadata. The layout is as follows: each message begins with a header of 6 data items followed by blocks of 4 items, one block for each receive data item.

1. ni5rcv	Message tag for control system receive message
2. ni5rcv	Number of data items in first receive message
3. ni5rcv	Task identifier of task sending message

4. ni5rcv	Control block number of ‘extfnctn’ control block receiving message
5. ni5rcv	Data format flag for acknowledgement
6. nr5rcv	Wait time for message
7. na5rcv	Name of first interactive variable to store first received data item
8. ni5rcv	Component number of interactive variable
9. ni5rcv	File index of ‘file’ containing interactive variables
10. ni5rcv	Offset of ‘file’ for interactive variable used to store first reveive data item
11. na5rcv	Name of interactive variable to store second receive data item

......